# 6 Parte VI: Da Válvula ao Transistor

A década de 1950 viu o surgimento da computação comercial. De um lado, a Remington Rand, que adquiriu a empresa de Mauchly e Eckert, e passou a produzir os UNIVAC comercialmente; de outro lado a IBM, iniciando com o modelo 701 e depois o 704, que foi um grande sucesso. Na Inglaterra, a Ferranti passou a produzir comercialmente os sucessores do Manchester Mark I.

Em termos de hardware, essa década viu a passagem gradual dos computadores a válvula para computadores transistorizados. As memórias dos computadores evoluíam dos tubos CRT e mercúrio para as memórias de núcleo magnético, que podiam ser muito maiores, rápidas e confiáveis a um custo relativamente baixo.

Em termos de software, a década viu o surgimento das primeiras linguagens de alto nível, em especial FORTRAN e LISP, que ainda hoje são usadas, e o ALGOL, que foi inspiração para a maioria das linguagens de programação usadas hoje em dia. Nos anos 1950 foram também desenvolvidos os primeiros jogos eletrônicos, embora ainda apenas como curiosidade, e não como a lucrativa indústria que são hoje. Os anos 1950 foram também os anos nos quais os computadores começaram a ser usados para tocar músicas e para digitalizar imagens.

## 6.1 Human Use of Human Beings – 1950

Considerando o quão incipiente era a computação em 1950 é de se admirar que tenham existido visionários tão fantasticamente influentes em nossa história quanto Norbert Wiener (Estados Unidos, 1894-1964). Depois de redefinir o termo “cibernética” em seu livro de 1948 ele brindou o mundo com uma nova e inspiradora visão de um futuro no qual a sociedade passa a ser uma mescla de homens e máquinas, ao publicar em 1950 o livro “The Human Use of Human Beings”[1].

Wiener estabelece que se os séculos XVII e XVIII foram a era dos relógios e o Século XIX a era das máquinas a vapor, o Século XX deu início a era da comunicação e controle.

Apesar de ter trabalhado em projetos militares durante a Segunda Guerra Mundial, Wiener recusou-se a continuar esse tipo de trabalho durante a guerra fria. Ele preferiu pesquisar e apresentar novos modos de organizar a sociedade.

É interessante notar a influência que seus pensamentos tiveram sobre o desenvolvimento posterior da computação visto que ela efetivamente nasceu a partir dos esforços relacionados à Guerra. Colocar a computação para trabalhar para a consolidação de uma nova sociedade baseada em informação e controle era algo novo. Com exceção de poucos pensadores como Vannevar Bush e Norbert Wiener, muitos viam os computadores apenas como “mastigadores de números”, que poderiam ser usados em poucas aplicações além dos cálculos das tabelas que somente teriam aplicações militares, financeiras e científicas.

[1] Tradução: o uso humano dos seres humanos.

Wiener considerava que a partir daquele momento da história, a integração dos seres humanos à tecnologia era inevitável. Ele disse: "*We have modified our environment so radically that we must now modify ourselves in order to exist in this new environment.*"[2].

## 6.2 Simon, o primeiro kit de computador pessoal – 1950

Se perguntarmos a qualquer profissional de computação hoje quando foi inventado o computador pessoal, a maioria provavelmente responderá que foi na década de 1970 e lembrará de máquinas como o Altair ou o Xerox Alto. Porém em 1950 já existia um computador pessoal, o Simon. Ele era anunciado como o "menor cérebro eletrônico do mundo" e qualquer pessoa podia comprar um kit e montá-lo em casa.

Edmund C. Berkeley (Estados Unidos, 1909-1988), criador do Simon, chegou a conhecer alguns computadores a partir de 1939, como o Complex Number Computer e o Harvard Mark I. Desde 1934 ele trabalhava para a companhia de seguros Prudential e chegou a escrever a especificação de um computador UNIVAC para a Prudential que foi encomendado à empresa de Eckert e Mauchly. Mas em 1949 a empresa o proibiu de participar de qualquer projeto que fosse contra a guerra atômica, mesmo em seu tempo livre, o que o fez pedir demissão. Passou então a trabalhar como consultor independente.

Logo depois disso ele escreveu um livro que o deixou famoso "Giant Brains, or Machines That Think"[3]. No livro ele descreve de forma acessível ao grande público como funcionavam essas máquinas elétricas e mecânicas. Em um determinado ponto ele diz, prevendo o futuro, que um dia o usuário dessas máquinas poderia escrever algo como "receita de biscoitos" e a máquina iria acessar um longo rolo de filme até chegar ao ponto exato no qual exibiria em uma tela três ou quatro livros com receitas de bolos. Sem saber, ele estava prevendo a existência dos atuais mecanismos de busca da Internet.

Neste mesmo livro ele propõe a construção de um "mini cérebro eletrônico", que ele chamou de "Simon". Ele deveria ser mais ou menos do tamanho de uma caixa de produtos de supermercado. Esse projeto, até onde se sabe, é a mais antiga concepção de computador pessoal, visto que desde Babbage até os dias dele, todos os computadores eram máquinas gigantescas que, uma vez construídos, jamais sairiam de suas salas inteiros. O Simon podia ser carregado com uma única mão, desde que a fonte fosse carregada com a outra.

Planos detalhados para a construção do Simon foram publicados em uma série de artigos na revista "Radio–Electronics". A capa original da revista com uma foto do computador pode ser visualizada em http://www.vintagecomputer.net/simon.cfm. Ele teria uma arquitetura semelhante à do Harvard Mark 1, com 129 relês e programação por fita perfurada de 5 colunas (tamanho padrão para os teletipos da época).

---

[2] Tradução: Nós modificamos nosso ambiente tão radicalmente que precisamos agora modificar a nós mesmos para existir nesse novo ambiente.
[3] Tradução: Cérebros Gigantes, ou Máquinas que Pensam.

Os registradores eram ora de 2 ora de 4 bits e a unidade de processamento era de 2 bits. As saídas eram visualizadas em um painel com 5 lâmpadas. Apesar de trabalhar apenas com números binários de dois bits, podendo representar assim apenas os números de 0 a 3, ele era capaz de realizar a adição e negação de números, verificar qual o maior dentre dois números, além de outras operações.

Berkeley não esperava que a máquina tivesse qualquer propósito comercial ou científico. Para ele ela era como um kit de experiências com o qual alguém poderia aprender sobre o funcionamento de cérebros eletrônicos. Era assim, um instrumento que aproximava a nova tecnologia que estava surgindo da pessoa comum. Um kit de construção do Simon custava cerca de 600 dólares, e até 1959 mais de 400 kits foram vendidos.

Em 1950, ainda, Berkeley fundou e editou uma revista que no ano seguinte se tornaria a Computers and Automation, possivelmente a primeira revista de computação no mundo.

## 6.3 Ferranti Mark I, o primeiro computador comercial – 1951

O Ferranti Mark I (Error: Reference source not found), produzido na Inglaterra, sobre o qual já falamos um pouco, foi o sucessor do Manchester Mark I. Ele é considerado o primeiro computador comercial do mundo efetivamente produzido. A primeira unidade foi entregue à Universidade de Manchester em fevereiro de 1951, enquanto que seu competidor americano, o UNIVAC, só foi vendido em 31 de março do mesmo ano e entregue apenas no final do ano seguinte.

Assim como o seu antecessor, o Ferranti também tinha memória principal baseada em tubos Williams, mas adicionou a esta uma memória secundária baseada em tambores (cilindros) magnéticos.

Outro avanço que esta máquina obteve foi um circuito multiplicador cerca de cinco vezes mais rápido do que o de seu antecessor. Alec Robinson projetou e construiu um circuito que fazia a multiplicação em paralelo, isso é, ao invés de multiplicar um bit depois do outro, esse circuito conseguia multiplicar vários bits ao mesmo tempo, acelerando assim o processo.

O Ferranti tinha uma característica interessante: um alto falante ligado aos seus circuitos, que emitia um som cuja tonalidade podia ser modificada através da programação da máquina. Não demorou até que os técnicos fizessem ele tocar algumas músicas reconhecíveis. De fato, a música foi até gravada e permanece como o mais antigo registro gravado de música tocada por computador. Porém, essa não foi a execução mais antiga, porque o CSIRAC, um computador australiano sobre o qual ainda falaremos, havia tocado músicas antes do Ferranti, embora não tenham sido gravadas na época.

O Ferranti era programado de uma maneira muito pouco intuitiva. Ele usava um código de base 32 (5 bits) para representar valores de 0 a 31. Os engenheiros resolveram manter o mesmo mapeamento entre os valores binários de 00000 a 11111 com

caracteres de acordo com o mapeamento do teclado da máquina. Assim, os “dígitos” de 0 a 31 seriam representados pela seguinte sequência:

| / | E | @ | A | : | S | I | U | ½ | D | R | J | N | F | C | K | T | Z | L | W | H | Y | P | Q | O | B | G | “ | M | X | V | £ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 |

Dessa forma, o número 4, por exemplo seria representando por “:”. Já o número 20 seria representado por “H” e 32 por “E/”. Os comandos da máquina também eram representados pelos mesmos caracteres. Daí se pode concluir que os programas do Ferranti eram verdadeiras “sopas de letrinhas”.

Apesar disso, ele era usável. Um de seus programadores chegou até a desenvolver um jogo limitado de xadrez para o qual, se houvesse xeque-mate em até 2 jogadas a máquina conseguia, examinar todos os movimentos possíveis e escolher aquele que a levaria à vitória. Esse programa é considerado o mais antigo jogo implementado em um computador de propósito geral.

## 6.4 UNIVAC – 1951

O UNIVAC ou Universal Automatic Computer foi o sucessor do ENIAC. Ele foi o primeiro computador comercial a ser vendido nos Estados Unidos. A máquina foi projetada e sua construção iniciada pela empresa Eckert-Mauchly Computer Corporation. Mas o projeto só foi concluído depois que Eckert e Mauchly venderam a empresa para a Remington Rand. Após o surgimento de outras versões da máquina, a versão original passou a ser conhecida como UNIVAC I.

O primeiro UNIVAC foi encomendado pelo escritório do Censo Americano em 1951. Já a quinta máquina pertencia à Comissão Americana de Energia Atômica. Em 1952, em uma jogada de marketing, a Remington pediu à rede de televisão CBS que ela usasse a máquina para prever o resultado da eleição presidencial daquele ano. Os especialistas previam uma vitória apertada de Adlai Stevenson, candidato Democrata. Mas o UNIVAC, com uma amostra de apenas 1% da população votante, previu uma ampla vitória para Dwight Eisenhower. Em função dessa discrepância, a CBS e a Remington preferiram não divulgar resultado da previsão do computador, o que teria sido feito em escala nacional, caso os resultados tivessem sido aceitos. E os programadores do UNIVAC possivelmente foram procurar as falhas no código ou nos dados. Porém, após a eleição de Eisenhower por uma ampla margem (442 a 89), viu-se que o computador de fato tinha acertado. Então a história foi divulgada.

Mauchly e Eckert criaram o UNIVAC especificamente para ser uma máquina para gerenciamento de empresas, não para fins científicos. Assim, ela não precisava de muitas das funções científicas que foram implementadas em máquinas anteriores, especialmente as do tempo da guerra. A Remington Rand, assim, com o UNIVAC, competia diretamente com a IBM, que até então dominava o mercado de automatização de empresas com suas máquinas tabuladoras.

Mas a Remington não chegou a ameaçar o domínio da IBM, pois faltava ao UNIVAC uma coisa muito importante: boas leitoras e gravadoras de cartões perfurados. As empresas já tinham muita informação armazenada nos cartões IBM e dificilmente

estariam dispostas a converter tudo isso em novos formatos como a fita magnética que o UNIVAC fornecia.

Embora inicialmente Eckert e Mauchly previssem que o UNIVAC custaria cerca de 150 mil dólares, os custos finais acabaram ficando na casa de 1,2 a 1,5 milhões. Ainda assim, 46 UNIVAC foram vendidos pela Remington.

A Figura Parte VI: Da Válvula ao Transistor -1 mostra o complexo painel de controle do UNIVAC, que mais parece uma cabine de um antigo avião a jato.


**Figura Parte VI: Da Válvula ao Transistor -1: Painel de controle do UNIVAC.[4] 1/1**

Ao todo o UNIVAC tinha 5200 válvulas, medindo 4,3 por 2,4 metros com uma altura de 2,6 metros. Ele pesava cerca de 13 toneladas e consumia 125 mil Watts.

Ele tinha uma memória grande para a época, com 1000 palavras de 12 caracteres. Essas 1000 palavras eram armazenadas em tubos de mercúrio semelhantes aos do EDVAC. Eram 100 tubos, cada um capaz de armazenar 10 palavras de 12 caracteres. Um dos módulos de memória de mercúrio do UNIVAC é mostrado na Figura Parte VI: Da Válvula ao Transistor -2.

[4] "Univac I at CHM.agr" by ArnoldReinhold - Own work. Licensed under CC BY-SA 3.0 via Commons - https://commons.wikimedia.org/wiki/File:Univac_I_at_CHM.agr.jpg#/media/File:Univac_I_at_CHM.agr.jpg

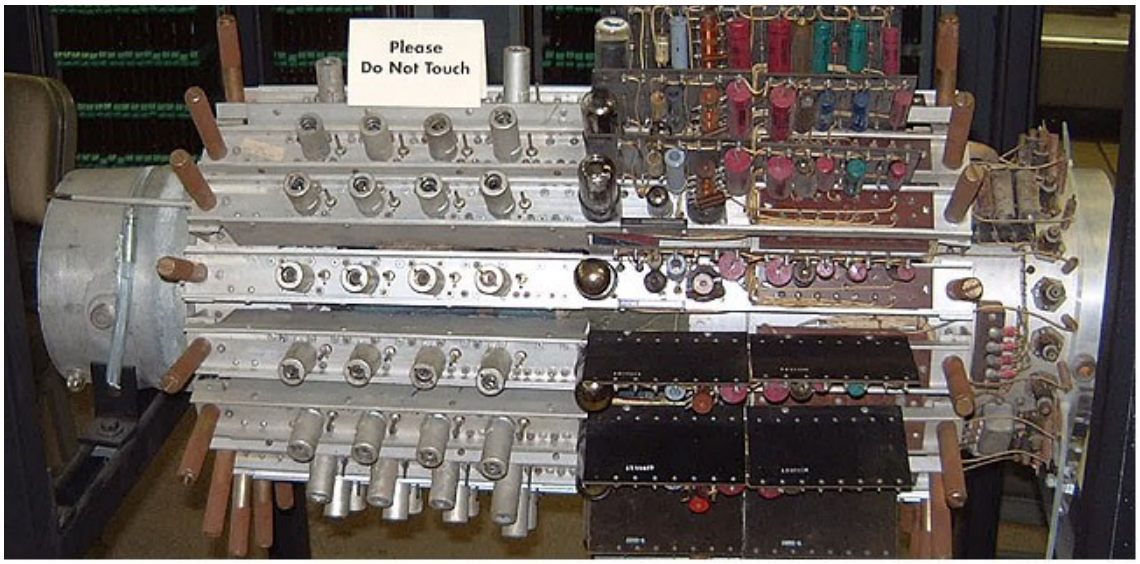


**Figura Parte VI: Da Válvula ao Transistor -2: Memória de mercúrio do UNIVAC.**[5] 1/1

Na tentativa de resolver os problemas de compatibilidade com os cartões perfurados da IBM, a Remington criou duas máquinas para converter cartões em fita magnética e vice-versa. Conta-se que uma destas máquinas tinha um compartimento dentro do duto de refrigeração que podia ser acessado por uma portinhola e que serviria, a pedido de um dos primeiros programadores do UNIVAC, para gelar cerveja quando ele estivesse trabalhando a noite.

## 6.5 LEO I – 1951

Praticamente ao mesmo tempo que a Remington lançava o UNIVAC na América, o Reino Unido também lançava um computador para uso empresarial. O LEO I, ou Lyons Electronic Office I foi construído por uma empresa de alimentos, a J. Lyons and Co. Após a guerra, em 1947, a empresa enviou dois de seus gerentes sênior, Oliver Standingford e Raymond Thompson para os Estados Unidos para conhecerem novos métodos de negócio desenvolvidos lá. Eles tiveram contato com o ENIAC de Mauchly e Eckert e rapidamente perceberam seu potencial para ajudar na administração de negócios. Ficaram sabendo também que em Cambridge, um outro computador eletrônico, o EDSAC, estava sendo construído.

Assim, em visita a Cambridge, a empresa decidiu primeiramente investir para ajudar a Universidade a finalizar o EDSAC, que foi finalizado em 1949. Após o projeto a empresa decidiu construir seu próprio computador, que seria uma expansão do EDSAC. Entre outras coisas, a Lyons usou o LEO I para processar sua folha de pagamento, controle de patrimônio, controle de vendas e *call center*. Pode-se afirmar que foi a primeira vez na história que uma única máquina integrou significativamente o gerenciamento de uma empresa.

---

[5] "Mercury memory". Licensed under CC BY-SA 3.0 via Commons - https://commons.wikimedia.org/wiki/File:Mercury_memory.jpg#/media/File:Mercury_memory.jpg

## 6.6 CSIRAC – 1951

Lembram que no Século XIX Ada Lovelace especulou que eventualmente os computadores poderiam compor e tocar músicas desde que instruídos com a regras da harmonia? Pois bem, essa visão se realizou pela primeira vez em 1951.

E surpreendentemente isso aconteceu na Austrália, onde foi construído um computador chamado CSIRAC, ou Commonwealth Scientific and Industrial Research Organisation Automatic Computer, inspirado na arquitetura do Manchester Baby. Ele continua intacto e é reputado como o mais antigo computador de primeira geração ainda existente. A Figura Parte VI: Da Válvula ao Transistor -3 é uma foto dele tirada em 2008.

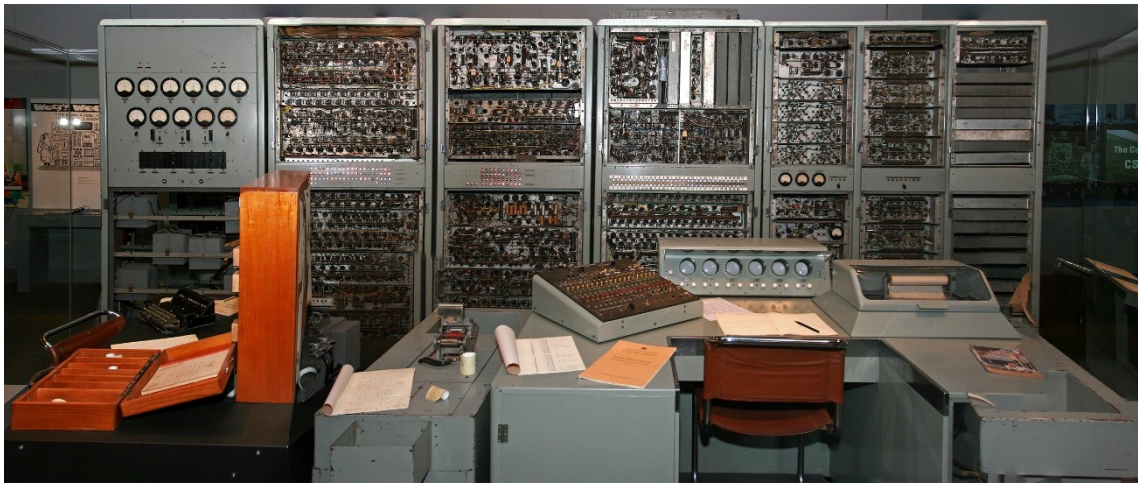

**Figura Parte VI: Da Válvula ao Transistor -3: CSIRAC.[6] 1/1**

Ele foi desenvolvido pelo Conselho de Pesquisa Científico e Industrial (CSIR) da Austrália. Ele foi desenvolvido por Trevor Pearcey (Reino Unido, 1919-1998) e Maston Beard. De acordo com os construtores, ele rodou o primeiro programa, para multiplicar dois números, no final de 1949. Porém nenhum deles lembra do dia exato; dizem que eles apenas gritaram “hurra!” e voltaram a trabalhar.

Ao contrário da maioria dos computadores, que transferem todos os bits de uma palavra em paralelo, através de vários condutores, o CSIRAC transmitia um bit de cada vez. Esses meios de condução são hoje chamados de “barramentos”. Assim, o CSIRAC tinha um barramento de 1 bit. O computador usava também uma memória de tubo de mercúrio e assim, de fato apenas um bit de cada vez era lido em cada tubo. Os construtores aproveitaram essa característica do CSIRAC para ligar um alto-falante a um dos barramentos, o qual produzia uma tonalidade sonora cada vez que uma certa configuração de bits passava por ali. Assim, eles foram capazes de programar o computador de forma que as tonalidades produzidas por ele se parecessem com músicas conhecidas. Desta forma, o CSIRAC além de ser o primeiro computador do hemisfério sul, também já foi o primeiro computador a cantar.

---

[6] "CSIRAC-Pano,-Melb.-Museum,-12.8.2008" by jjron - Own work. Licensed under CC BY-SA 3.0 via Wikimedia Commons - https://commons.wikimedia.org/wiki/File:CSIRAC-Pano,-Melb.-Museum,-12.8.2008.jpg#/media/File:CSIRAC-Pano,-Melb.-Museum,-12.8.2008.jpg

## 6.7 MIT Whirlwind, o primeiro computador de tempo real – 1951

O Whirlwind (tornado), mostrado na Figura Parte VI: Da Válvula ao Transistor -4, foi um computador a válvula desenvolvido pelo MIT (Massachusetts Institute of Technology) para a marinha americana. Ele foi o primeiro computador na história a ser projetado para trabalhar em tempo real. Até então todos os computadores que existiam processavam dados de entrada e produziam uma saída. Assim, se o computador fosse mais rápido, ótimo, e se fosse mais lento era só o caso de esperar um pouco mais para obter os resultados. Esse tipo de processamento é até hoje chamado de processamento *batch*, ou "em lote".

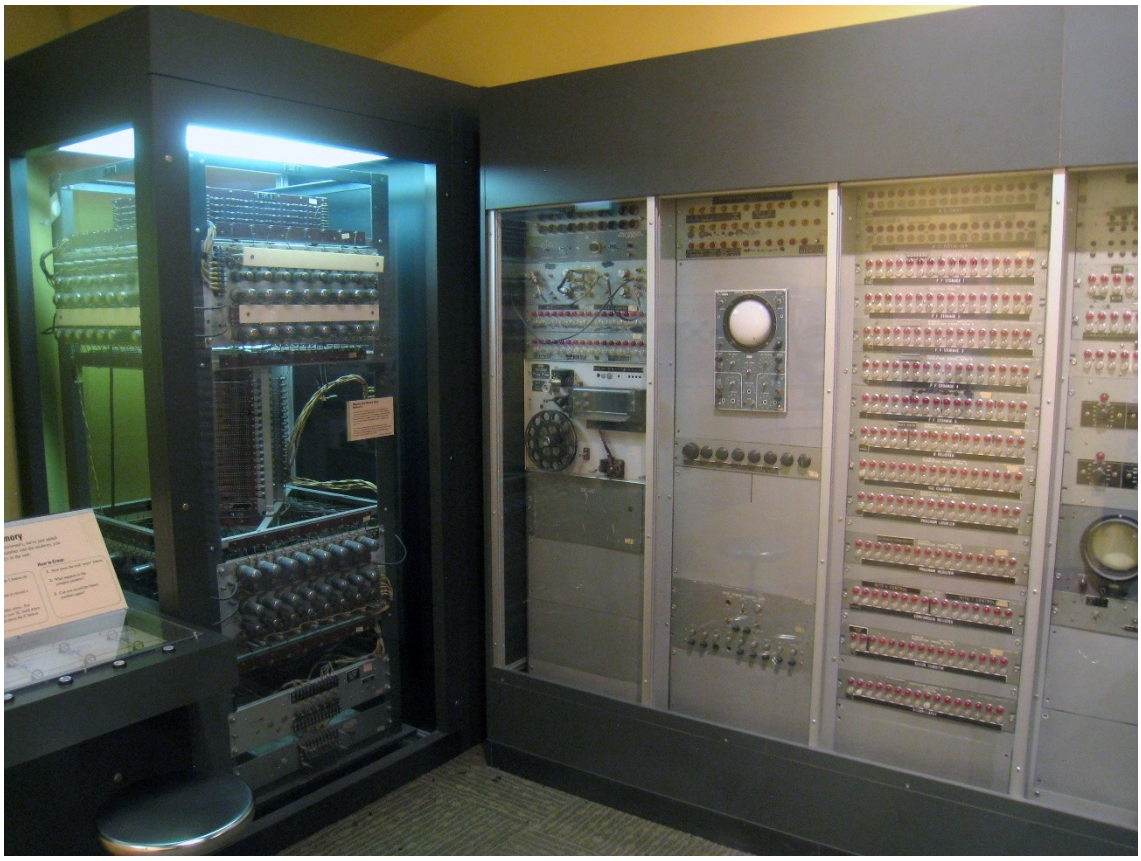

**Figura Parte VI: Da Válvula ao Transistor -4: Whirlwind.**[7] **1/1**

Mas o Whirlwind foi pensado para outro tipo de aplicação. Eles queriam um computador que pudesse controlar os parâmetros de um simulador de voo. Mais do que isso: simuladores de voo baseados em computação analógica já existiam naquela época, mas eles queriam um simulador que pudesse simular qualquer tipo de aeronave. Apenas um verdadeiro computador digital poderia fazer isso; a partir de dados de configuração predefinidos ele poderia simular diferentes aeronaves. Mas este computador teria que ser muito rápido, pois seria o primeiro a operar em tempo real, ou seja, não se poderia esperar pelo resultado de um processamento: cada ação do piloto sobre o simulador teria que surtir efeito imediatamente.

---

[7] "Museum of Science, Boston, MA - IMG 3168" by Daderot - Own work. Licensed under Public Domain via Commons - https://commons.wikimedia.org/wiki/File:Museum_of_Science,_Boston,_MA_-_IMG_3168.JPG#/media/File:Museum_of_Science,_Boston,_MA_-_IMG_3168.JPG

Depois de experimentar com arquiteturas analógicas durante a guerra, Jay W. Forrester (Estados Unidos, 1918), encarregado do projeto, concluiu que esse tipo de computador era muito pouco preciso e pouco flexível. Assim, em 1945, após a equipe ter tido contato com o ENIAC, chegaram à conclusão que uma máquina digital seria a solução.

O problema é que as memórias de mercúrio que então estavam em uso na maioria dos computadores eram muito lentas para os propósitos do Whirlwind. Eles acabaram decidindo por usar tubos CRT, ou tubos Williams, que eram uma opção mais rápida. Porém, a equipe considerou que os ciclos de reforço de sinal que eram necessários para manter a memória iriam atrasar os tempos de processamento; como vimos, o sinal eletrostático na tela do CRT permanecia por apenas um quinto de segundo, e era necessário reforçar todos os pontos da tela antes desse tempo passar para evitar que a memória simplesmente se apagasse. E isso tomava tempo de processamento.

Eles resolveram apostar em uma nova tecnologia de tubos eletrostáticos, que estava sendo desenvolvida no MIT que tornaria desnecessários os ciclos de reforço de sinal. Haveria um segundo cátodo no tubo que enviaria uma enxurrada de elétrons de baixa energia para a tela inteira. A técnica era chamada de “inundação” e, se funcionasse, os pontos carregados positiva ou negativamente na tela ficariam ali permanentemente, sem necessidade de ciclos de reforço.

Mas infelizmente a nova tecnologia não funcionou. Desesperado, Forrester procurava por alternativas, até se deparar com uma propaganda de uma companhia que estava desenvolvendo núcleos magnetizáveis que ele percebeu imediatamente que seriam uma alterativa para a construção da memória para o Whirlwind.

Nasceu assim a memória de núcleo magnético, que foi o padrão da indústria de computadores até 1975. A Figura Parte VI: Da Válvula ao Transistor -5 mostra esquematicamente uma memória de núcleo magnético de 16 bits, organizados em 4 linhas de 4 colunas. Cada núcleo é um anel de material magnetizável que ao ser exposto a uma corrente elétrica que passa por dentro do anel pode ser magnetizado em uma direção ou em outra, assim, pode-se considerar que um anel magnetizado em sentido horário armazena um 0 e um anel magnetizado em sentido anti-horário armazena um 1.

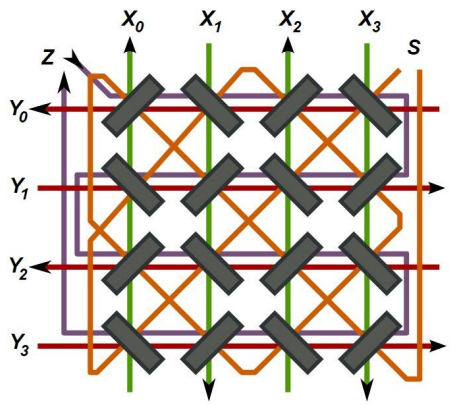


**Figura Parte VI: Da Válvula ao Transistor -5: Esquema de uma memória de núcleo magnético.**[8] 1/3

Os anéis, depois de magnetizados, conseguem manter este estado indefinidamente, ou seja, é uma memória não-volátil, pois mesmo que a corrente seja totalmente desligada, as informações ali armazenadas são mantidas. Ocorre que a cerâmica da qual são feitos os anéis só muda seu sentido de magnetismo se for exposta a um certo nível de corrente. Enquanto a corrente estiver abaixo deste nível, não há mudança. Assim, um engenhoso mecanismo matricial permite magnetizar qualquer um dos anéis individualmente.

Observe na Figura Parte VI: Da Válvula ao Transistor -5 que há quatro fios $X_0$ ... $X_3$ e mais quatro fios $Y_0$ ... $Y_3$. Se você quiser armazenar, por exemplo, 1 no núcleo na posição (2, 2), o que tem a fazer é passar metade da corrente necessária para magnetizar um núcleo por X2 e metade da corrente necessária em Y2. Assim, apenas o núcleo (2, 2) será exposto a uma corrente suficientemente alta para mudar seu estado, enquanto que os outros núcleos permanecem no estado que estavam antes.

As linhas S e Z, mostradas na Figura Parte VI: Da Válvula ao Transistor -5 passam por todos os núcleos e são respectivamente o “sensor” e o “inibidor” da memória. Versões posteriores passaram a usar apenas uma linha SZ visto que embora tenham funções diferentes, elas são sempre usadas em momentos diferentes.

A linha S serve, portanto, como sensor, ou seja, é ela que permite fazer a leitura de uma dada posição de memória. A leitura em uma memória de núcleo magnético destrói a informação ali armazenada, ou seja, quando a informação é lida ela é destruída. Isso ocorre porque o processo de leitura consiste em tentar induzir o valor 0 em um determinado núcleo, digamos (2, 2). Assim, a leitura é feita aplicando-se metade da corrente, como vimos antes, nos fios X2 e Y2, tentando mudar o valor do núcleo (2, 2) para zero. Se (2, 2) já está em zero, nada ocorre, mas se (2, 2) está em 1, haverá uma inversão de sua polarização e essa inversão causará um pulso no fio S, o qual poderá ser lido.

---

[8] "Coincident-current magnetic core" by Tetromino - Own work / собственная работа. Licensed under CC BY 3.0 via Commons - https://commons.wikimedia.org/wiki/File:Coincident-current_magnetic_core.svg#/media/File:Coincident-current_magnetic_core.svg

Há um tempo de espera para esse pulso. Se aplicarmos a corrente em X2 e Y2 e após o tempo de espera não houver pulso nenhum em S é porque (2, 2) já contém 0. Porém, se houver o pulso é porque (2, 2) continha um 1.

Mas agora, depois da leitura, (2, 2) estará com zero. Então, se seu estado anterior fosse “1” será necessário regravar o “1” na posição (2, 2) aplicando novamente a meia corrente (agora inversa) em X2 e Y2.

Para a escrita, parte-se do princípio que se a posição de memória foi lida, então ela está com um “0”. Para escrever “1” aplica-se meia corrente em X e Y, como vimos antes. Para escrever um “0”, ou melhor, para evitar que o núcleo deixe de ser zero, aplica-se meia corrente inversa na linha do inibidor (Z). Assim, a corrente total que passará pelo núcleo será metade da corrente necessária para invertê-lo e assim, ele permanecerá com um “0” armazenado ali.

A Figura Parte VI: Da Válvula ao Transistor -6 mostra uma foto de uma memória de núcleo magnético real. Os espaços entre os núcleos é de cerca de 1mm.

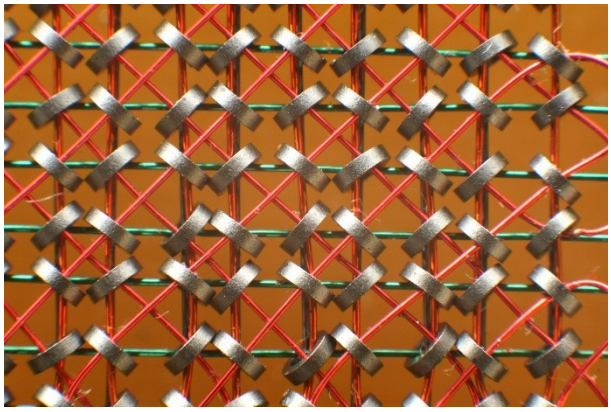

**Figura Parte VI: Da Válvula ao Transistor -6: Memória de núcleo magnético.[9] 1/2**

## 6.8 OXO – 1952

O primeiro jogo de computador gráfico foi implementado em 1952 por Alexander S. Douglas (Reino Unido, 1921-2010) durante seu doutorado em Cambridge. Lá ele teve contato com o computador EDSAC. O EDSAC tinha memória baseada em tubos de mercúrio, mas o conteúdo da memória podia ser exibido em um tubo CRT.

Assim, Douglas implementou o OXO, ou “jogo da velha”, desenhando a armação do jogo na tela e implementando um algoritmo de decisão (uma forma primitiva de inteligência artificial) para fazer um jogo perfeito contra um humano. Assim, ou a máquina ganhava ou empatava.

O humano podia entrar com sua jogada através de um disco de telefone, no qual ele selecionava uma posição entre 1 e 9. Um pequeno filme com uma simulação do jogo pode ser visto em https://www.youtube.com/watch?v=SRnGIzn8DA0.

---

[9] "KL Kernspeicher Makro 1" by Konstantin Lanzet - received per EMailCamera: Canon EOS 400D. Licensed under CC BY-SA 3.0 via Commons - https://commons.wikimedia.org/wiki/File:KL_Kernspeicher_Makro_1.jpg#/media/File:KL_Kernspeicher_Makro_1.jpg

## 6.9 Compilador A-0 – 1952

Grace M. Hopper (Estados Unidos, 1906-1992) foi uma destacada programadora do Harvard Mark I. Já nos referimos a ela por ter sido a primeira pessoa a realmente encontrar um "bug" em um computador. Em 1949 Eckert e Mauchly contrataram Hopper para a equipe de desenvolvimento de software para o UNIVAC.

**Figura Parte VI: Da Válvula ao Transistor -7: Grace Hopper operando o UNIVAC.**[10] **1/2**

Em 1952 ela finalizou o projeto do Sistema A-0, considerado o primeiro programa compilador efetivo, isto é, um programa que traduz instruções mais abstratas em comandos que a máquina pode executar. Teria sido ela, inclusive, a pessoa a cunhar o termo "compilador".

Seu compilador era bem simples, na verdade, era mais parecido com um carregador de rotinas. Ela selecionou um conjunto de programas que realizavam cálculos específicos, por exemplo, seno, cosseno, etc., e colocou todos esses programas em uma fita. Ela anotou a posição inicial de cada um destes programas na fita de forma que pudesse fazer a máquina avançar para a posição correta sempre que precisasse de um deles.

Assim, um "programa" escrito em A-0 nada mais era do que uma lista de chamadas para essas rotinas, cada chamada era representada por um número que correspondia à uma posição da fita e após esse número eram passados os valores dos parâmetros.

O A-0, assim, não era um compilador na acepção moderna do termo. Mas, esse projeto motivou Hopper a continuar trabalhando em seu aperfeiçoamento. A área de pesquisa foi chamada de "programação automática"; hoje nós a chamamos de "programação em alto nível", para diferenciar da programação em código de máquina. Programação automática seria assim a possibilidade de programar o computador com uma linguagem semelhante à linguagem natural, como inglês ou português.

Em um artigo publicado em 1952, Richard Ridgway apresenta um comparativo de tempo dispendido por programadores usando o A-0 e usando o método de

[10] By Unknown (Smithsonian Institution) - Flickr: Grace Hopper and UNIVAC, CC BY 2.0, https://commons.wikimedia.org/w/index.php?curid=19763543

programação convencional. Com a técnica usual foram necessários 740 minutos de programação, contra 20 minutos usando o compilador.

O A-0 foi sucedido pelo A-1, A-2 e A-3. Conta-se que a Remington Rand disponibilizou o código fonte (programa) do A-2 gratuitamente para os usuários de seus computadores UNIVAC e que também os encorajava a enviarem sugestões e feedback sobre o sistema. Possivelmente essa foi então a primeira iniciativa de software livre e de código aberto a existir no mundo. A partir do A-2 a programação já era feita usando uma notação que eles chamaram de pseudocódigo, ou seja, um código que a máquina não conseguiria processar normalmente. Mas eles construíram um sistema que lia as instruções em pseudocódigo da fita magnética e as convertia em instruções que a máquina podia executar.

As versões A-0 até A-3 eram chamadas de ARITH-MATIC, uma versão posterior, AT-3, foi chamada de MATH-MATIC e a versão final, B-0 de FLOW-MATIC. A FLOW-MATIC, uma linguagem já muito parecida com inglês estruturado, foi a base para a criação da linguagem COBOL.

Comenta-se que Hopper teria enfrentado grande resistência por parte daqueles que programavam com código de máquina. Ela chegou a desabafar: "*I had a running compiler, and nobody would touch it because, they carefully told me, computers could only do arithmetic; they could not do programs.*"[11].

## 6.10 IBM 701 – 1952

A IBM não ficou restrita apenas ao mercado de computadores comerciais. Seu primeiro grande projeto de um computador científico para produção em escala, inicialmente chamado de Defense Calculator, foi lançado em 1952 com o nome de IBM 701 (Figura Parte VI: Da Válvula ao Transistor -8).

[11] Tradução: Eu tinha um compilador que funcionava e ninguém queria tocar nele porque, eles me falavam com muito cuidado, computadores poderiam lidar apenas com aritmética, não com programas.


**Figura Parte VI: Da Válvula ao Transistor -8: Console do IBM 701.**[12] 1/2

Tratava-se de um computador a válvulas com memória baseada em 72 tubos Williams com capacidade de 1024 bits cada. Assim, a máquina tinha uma memória total de 2048 (2K) palavras de 36 bits. Dezenove computadores modelo 701 foram fabricados. O aluguel girava em torno de 8 mil dólares por mês. 99153 9050

O folclore atribui a Watson, presidente da IBM, a frase “eu acho que no mundo há mercado para no máximo cinco computadores”. Mas essa lenda provavelmente deve-se à afirmação de Watson de que antes da produção do 701 ele saiu para visitar 20 potenciais clientes na esperança de vender para pelo menos 5 deles, mas voltou com 18 pedidos.

Essa foi a primeira máquina totalmente eletrônica e com programa armazenado fabricada pela IBM. Uma das unidades do 701 substituiu o SSEC na Avenida Madison em Nova York. Ela era 25 vezes mais rápida e ocupava menos de 25% do espaço ocupado pelo SSEC, que era construído com relês. O 701 foi o marco para a mudança do foco da IBM do negócio com tabuladoras para o negócio dos computadores eletrônicos.

[12] "IBM 701console" by Dan - Flickr: IBM 701. Licensed under CC BY 2.0 via Wikimedia Commons - https://commons.wikimedia.org/wiki/File:IBM_701console.jpg#/media/File:IBM_701console.jpg

## 6.11 IAS – 1952

John von Neumann, professor de Princeton e consultor de Eckert e Mauchly durante algum tempo, também projetou e construiu em 1952 um computador, o IAS (Figura Parte VI: Da Válvula ao Transistor -9), nome dado em homenagem ao instituto onde ele foi construído, o Institute of Advanced Studies. O IAS é também conhecido como "Computador de von Neumann".

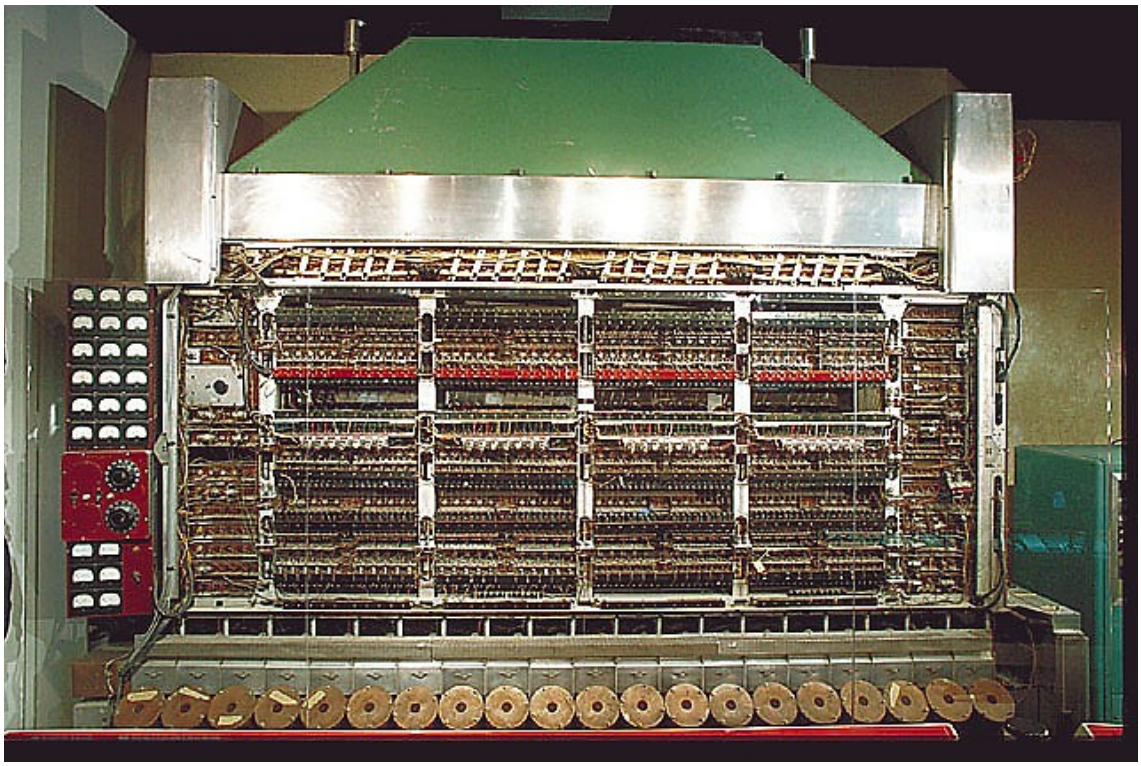

**Figura Parte VI: Da Válvula ao Transistor -9: IAS.[13] 1/1**

Ele não foi o primeiro computador a implementar a arquitetura de programa armazenado, pois o Manchester Baby já fazia isso anos antes. Mas von Neumann inovou mais uma vez ao identificar a possibilidade de realizar comandos de repetição em memória, ou seja, com o programa armazenado em memória não seria mais necessário colar as duas pontas de uma fita de programa para realizar instruções em ciclos repetitivos; isso poderia ser feito por comandos em memória.

## 6.12 Trackball – 1952

Quem foi inventado primeiro? O *mouse* ou a *trackball*? Bem, foi a trackball, e foi em 1952.

O primeiro protótipo de uma trackball, chamado "rollerball" foi criado por Ralph Benjamin em 1941 para substituir um sistema de joystick que seria usado para entrar com coordenadas para calcular trajetórias de projéteis. Ele achava o joystick

---

[13] "Princeton IAS computer" by National Museum of Americal History - http://americanhistory.si.edu/collections/search/object/nmah_334741. Licensed under Public Domain via Commons - https://commons.wikimedia.org/wiki/File:Princeton_IAS_computer.jpg#/media/File:Princeton_IAS_computer.jpg

desajeitado e propôs usar uma bola metálica encostada em duas rodas. Girando a bola, as rodas “leriam” as posições x e y. Ele patenteou a ideia em 1947, mas ela nunca foi colocada em prática.

Assim, a primeira trackball efetivamente construída e usada foi fabricada no Canadá, com uma bola de boliche canadense, que é menor do que a bola americana, mais parecida com a nossa bola do jogo de bolão.

Este equipamento foi usado como dispositivo de entrada em um projeto da Ferranti Canada, subsidiária da Ferranti britânica, responsável pela construção do Ferranti Mark I. O projeto em questão chamava-se DATAR, ou Digital Automated Tracking and Resolving, que seria um sistema de comunicação e processamento de dados eletrônicos navais para a Marinha Canadense. O objetivo do projeto era ser um sistema integrado que permitisse consultar e controlar a posição de navios e aviões em tempo real.

A trackball permite ao usuário mover a esfera para registrar uma dupla entrada de dados no computador. Com movimentos da mão sobre a esfera o usuário muda os eixos x e y, ou coordenadas de entrada. Há, além dos suportes para a bola, dois discos, colocados em posição ortogonal (90 graus um do outro). Ao girar a bola em qualquer sentido, giros correspondentes são capturados por esses discos e transmitidos ao computador.

O projeto DATAR, infelizmente não foi concluído da forma como se esperava, pois a máquina que faria o processamento dos dados já estava com mais de 30 mil válvulas, e os erros devidos à queima de válvulas estavam acima do aceitável. Anos depois, versões transistorizadas do sistema foram construídas.

Mas o DATAR deixou um interessante legado para o Canadá, pois o país não tinha ainda uma cultura de construção de computadores. Com o projeto DATAR a Ferranti Canadá passou a investir na educação de seus jovens engenheiros, que se tornaram depois pessoas-chave na construção da cultura eletrônica e computacional do país.

## 6.13 IBM 650 e 704 – 1953

O primeiro computador produzido em massa foi o IBM modelo 650, ou IBM Magnetic Drum Data-Processing Machine. Cerca de 2 mil cópias dele foram produzidas desde seu anúncio em 1953 até 1962. Os números no 650 eram representados de forma semelhante ao ábaco, em “bi-quinário”, com dois bits (o “bi”) para representar se o dígito era de 0 a 4 ou de 5 a 9, e mais cinco bits para representar o número em si (o “quinário”). Assim, os dígitos de 0 a 9 eram representados assim:

- 0: 10-10000
- 1: 10-01000
- 2: 10-00100
- 3: 10-00010
- 4: 10-00001
- 5: 01-10000
- 6: 01-01000

- 7: 01-00100
- 8: 01-00010
- 9: 01-00001

A Figura Parte VI: Da Válvula ao Transistor -10 mostra um detalhe do painel de controle do 650 onde claramente se veem os indicadores numéricos bi-quinários.


**Figura Parte VI: Da Válvula ao Transistor -10: Painel do IBM 650.[14] 1/1**

O 650 era vendido como uma máquina para aplicações científicas e de engenharia. Como era relativamente barato e versátil chegou a ser usado não apenas comercialmente, mas também como máquina para ensinar ciência da computação nas universidades. O 650 tinha uma memória de tambor magnético que podia conter 1, 2 ou 4 mil palavras (1, 2 ou 4 KB).

Donal Knuth (Estados Unidos, 1938), autor da influente série de livros "The Art of Computer Programming" de 1968, dedicou a série ao IBM 650: "*This series of books is affectionately dedicated to the Type 650 computer once installed at Case Institute of Technology, with whom I have spent many pleasant evenings.*"[15].

Em 1954 a IBM lança outro marco da história da computação, o modelo 704, que seria considerado como sucessor do 701, um computador eminentemente científico, capaz de operar com números de ponto flutuante. Uma das principais melhorias foi o uso de memória de núcleo magnético no lugar dos tubos Williams. Outra foi a adição de

[14] "IBM 650 panel close-up of bi-quinary indicators" by Mfc - Own work. Licensed under CC BY-SA 3.0 via Commons - https://commons.wikimedia.org/wiki/File:IBM_650_panel_close-up_of_bi-quinary_indicators.jpg#/media/File:IBM_650_panel_close-up_of_bi-quinary_indicators.jpg
[15] Tradução: Esta série de livros é afetuosamente dedicada ao Computador Modelo 650 que foi instalado no Instituto de Tecnologia Case, com o qual eu passei muitas noites agradáveis.

registradores de índice, que permitiram realizar operações sobre vetores de forma bem mais eficiente.

## 6.14 Primeiros Sistemas Operacionais – 1954

Os primeiros computadores eram simplesmente máquinas que rodavam programas previamente construídos, um de cada vez. O usuário chegava, introduzia o programa na máquina (as vezes arduamente mudando plugues e chaves, mas depois usando fitas e cartões perfurados), introduzia os dados, iniciava o programa e aguardava o final do processamento, com as saídas sendo impressas ou perfuradas no meio adequado.

Os computadores mais antigos não tinham nenhuma facilidade para, por exemplo, rodar vários programas, um após o outro, de forma automática. Havia sempre a necessidade de um operador humano fazer a retirada do programa anterior e a colocação do novo programa. Nessa época, a "fila de trabalho" dos computadores era literalmente uma fila de pessoas na porta da sala do computador, aguardando para rodarem seus programas. Comenta-se que na Universidade de Cambridge, na Inglaterra, os programas em fitas perfuradas eram colocados em varais de roupa e a cor do prendedor usado indicava a prioridade do programa na fila.

Porém, em meados dos anos 1950 os computadores começaram a ficar extremamente rápidos, em comparação com os anteriores e, assim, mais tempo se gastava fazendo a colocação dos programas na máquina do que efetivamente rodando os programas. Como os computadores eram máquinas absurdamente caras, esse tempo ocioso, em que elas não estavam rodando nenhum programa, tinha que ser minimizado.

Porque não construir um programa capaz de pegar vários programas em uma fila e automaticamente introduzi-los na máquina? Isso foi feito, e assim nasciam os primeiros sistemas operacionais. Na época eles eram chamados de "monitores" ou "programas monitores", e o tipo de trabalho que esses programas faziam é hoje equivalente ao que o kernel dos modernos sistemas operacionais faz.

O nome "sistema operacional" pareceu mais adequado porque ele realmente substitui o operador em muitas de suas tarefas. Com o método manual, um operador humano teria que ficar 24 horas por dia com a máquina para que ela pudesse operar, trocando os programas e dados. Mas com o sistema operacional, a máquina podia trabalhar sozinha. Bastava que os diversos programas fossem enfileirados na leitora de fita ou leitora de cartões e com as devidas diretivas (cartões de "job") a máquina conseguia sequenciar esses programas automaticamente, sem a necessidade do operador humano.

Outra coisa que esses sistemas operacionais logo começaram a incorporar foram as bibliotecas de rotinas que forneciam um acesso mais fácil para as diferentes partes do hardware ou periféricos da máquina. Nos primeiros tempos, um programador teria que saber exatamente quais comandos havia e como funcionava todo o hardware e seus periféricos. Se houvesse, por exemplo, três modelos de leitora de fita, cada um deles com comandos diferentes, o programador teria que conhecer todos eles e saber como

programá-los. Já com o sistema operacional, o programador usaria uma rotina mais abstrata, a qual faria o serviço na leitora de fita independentemente de quem fosse seu fabricante ou qual fosse seu conjunto de instruções. Esse tipo de rotina hoje é chamada de “driver” e é o que nos permite usar as mais diversas impressoras, scanners, etc., sem precisar saber programá-los.

De acordo com a literatura, os três primeiros sistemas operacionais foram desenvolvidos em 1954, 1955 e 1956. Em 1954, o MIT desenvolveu uma espécie de sistema operacional chamado Tape Director para o seu UNIVAC. Em 1955 A General Motors desenvolveu um sistema operacional para o seu IBM 701 e em 1956 desenvolveu um sistema chamado GM-NAA I/O para o seu IBM 704.

Nessa época os sistemas operacionais eram quase sempre desenvolvidos pelos clientes e não pelos fabricantes das máquinas, pois eram os clientes que sentiam na carne a necessidade de ter um sistema deste tipo.

Outra coisa que os sistemas operacionais proporcionaram desde muito cedo foi a possibilidade de multitarefas. Por exemplo, nos primeiros computadores, enquanto a impressora estivesse imprimindo, o computador ficava parado aguardando o final da impressão. Com a introdução dos sistemas operacionais foi possível também permitir que o computador começasse a rodar o programa seguinte enquanto o anterior ainda estivesse imprimindo sua saída. Essa técnica foi chamada de *spooling* e significa: “simutaneous peripheral operation on line”[16]. Mais tarde, até mesmo foi possível rodar vários programas ao mesmo tempo na mesma máquina.

## 6.15 Harwell Cadet, o primeiro computador totalmente transistorizado – 1955

Existe alguma disputa sobre qual teria sido o primeiro computador transistorizado. O fato é que em meados da década de 1950, mesmo com válvulas mais confiáveis do que as da década de 1940, elas ainda eram uma incomodação: custavam caro, queimavam com frequência, ocupavam muito espaço e consumiam uma quantidade absurda de energia.

O caminho seria investir em semicondutores sólidos e o grande candidato era o transistor, inventado em 1947. O transistor em si vinha evoluindo também, de versões de laboratório para versões mais aceitáveis para uso industrial.

Os computadores mecânicos, eletromecânicos e a válvula, são considerados como a Primeira Geração de computadores. Com o advento dos computadores transistorizados, inicia-se a Segunda Geração.

A primeira experiência de que se tem registro ocorreu na Universidade de Manchester, Reino Unido, com Tom Kilburn (Reino Unido, 1921-2001) onde um pequeno protótipo de computador transistorizado, o “University of Manchester's experimental Transistor Computer”, foi construído em 1953 e depois em tamanho real em 1955. Porém, essa

[16] Tradução: operação simultânea de periféricos em linha.

máquina ainda usava algumas válvulas para gerar pulsos de clock[17] mais rápidos e assim ela não é considerada como totalmente transistorizada.

Em 1954 os Laboratórios Bell nos Estados Unidos criaram uma máquina semelhante, o TRADIC, ou TRAnsistor DIgital Computer. Ele também usava válvulas para gerar pulsos de clock, já que nenhum transistor da época conseguia gerar toda a energia necessária (30 Watts) com a mesma velocidade.

A IBM anunciou em 1955, mas só lançou em 1957, uma calculadora totalmente transistorizada, a IBM 608, que continha mais de 3 mil transistores de germânio.

A máquina que pode ser considerada com propriedade como o primeiro computador de propósito geral totalmente transistorizado é o Harwell Cadet (Figura Parte VI: Da Válvula ao Transistor -11). Em 1953 a divisão de eletrônica das instalações de pesquisa em energia atômica de Harwell, Reino Unido, resolveram investir na tecnologia de transistores para substituir uma máquina mais antiga. Cadet é a sigla para Transistor Electronic Digital Automatic Computer de trás para frente. Ele rodou seu primeiro programa em fevereiro de 1955.

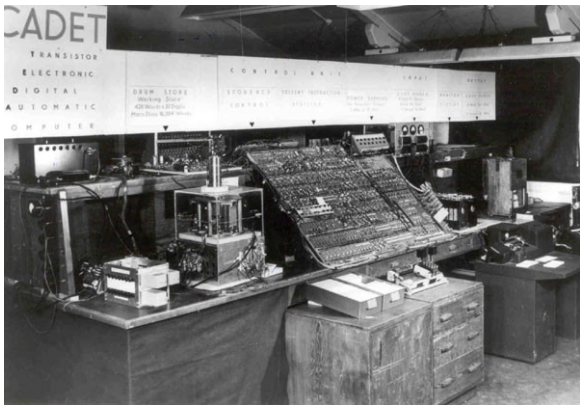


**Figura Parte VI: Da Válvula ao Transistor -11: Harwel Cadet.[18] 1/2**

Um dos preços que os construtores tiveram que pagar para ter uma máquina totalmente transistorizada foi um clock baixo, de apenas 58 KHz, quando computadores a válvula já tinham clock até 20 vezes mais rápido.

Em 1956, o Laboratório Lincoln do MIT também construiu um computador totalmente transistorizado, o TX-0 (apelidado de “ticso”). Ele utilizava um novo tipo de transistor de alta velocidade desenvolvido pela Philco em 1953. Com o sucesso da máquina, logo passaram a desenvolver um projeto mais sofisticado, o TX-1, mas como esse demonstrou ser complexo demais, partiram para outro conceito mais simples, o TX-2, que acabou sendo construído com partes do TX-0, já que partes de computadores, especialmente memórias eram demasiadamente caras naquela época.

---

[17] São os sinais que permitem que o computador sincronize a execução de suas instruções. A princípio, um ciclo de clock mais rápido implica que o computador vai executar suas instruções mais rapidamente também.

[18] "HarwellCadetComputer" by MichaelWilson78 - Own work. Licensed under Public Domain via Commons - https://commons.wikimedia.org/wiki/File:HarwellCadetComputer.jpg#/media/File:HarwellCadetComputer.jpg

As partes que sobraram do TX-0 acabaram sendo doadas para o Laboratório de Pesquisa Eletrônica do MIT, onde ele se tornaria peça chave para a criação do futuro Laboratório de Inteligência Artificial do MIT e da cultura hacker.

## 6.16 Disco Magnético – 1956

Até 1956 os dispositivos de armazenamento secundário dos computadores eram basicamente a fita perfurada ou magnética, os cartões perfurados e os tambores magnéticos. No caso das fitas e cartões existia um problema muito sério porque estes meios de armazenamento eram memórias lineares. Para atingir um determinado ponto da fita, por exemplo, você precisava avançar ou rebobinar até ele. Um programa que acessasse dados em posições variadas numa fita ficaria mais tempo avançando e rebobinando do que fazendo qualquer outra coisa, já que essas operações eram demoradas.

No caso dos tambores magnéticos, não havia esse problema; qualquer ponto do tambor podia ser acessado por um cabeçote em um tempo muito curto. Mas eles ocupavam muito espaço físico. Em 1956 a IBM inovou a indústria de armazenamento secundário ao lançar o primeiro sistema de armazenamento baseado em discos, o disco rígido ou hard drive como hoje é conhecido. A principal vantagem em relação às fitas e cartões era que qualquer dado armazenado podia ser acessado rapidamente, pois cabeçotes de leitura corriam sobre as 100 trilhas concêntricas de cada disco que girava a 1.200 rotações por minuto (bem mais que um LP, que usualmente girava a 33 RPM), e permitiam rapidamente alcançar uma grande quantidade de dados.

Em relação aos tambores, a vantagem dos discos era a economia de espaço, pois não se usaria apenas o cilindro externo para gravar dados, mas praticamente todo o interior do cilindro no qual vários discos eram empilhados, como mostrado na Figura Parte VI: Da Válvula ao Transistor -12.

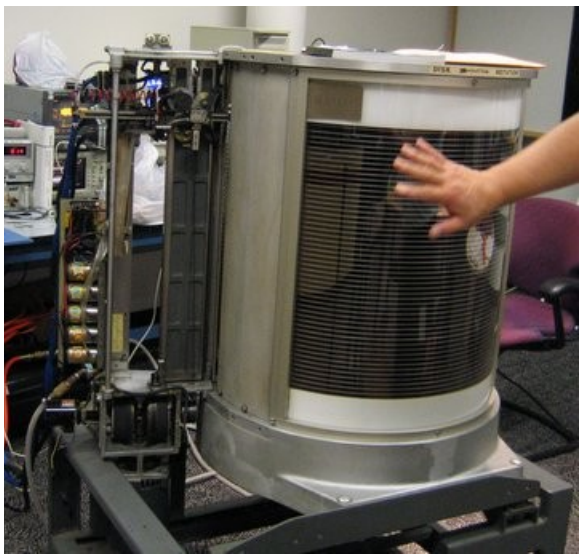

**Figura Parte VI: Da Válvula ao Transistor -12: IBM 350 RAMAC.**[19] **1/2**

[19] "IBM 350 RAMAC" by Transferred from en.wikipedia to Commons by DingirXul.. Licensed under CC BY-SA 2.5 via Commons - https://commons.wikimedia.org/wiki/File:IBM_350_RAMAC.jpg#/media/File:IBM_350_RAMAC.j

Os primeiros computadores a terem esse sistema foram o IBM 305 RAMAC e o IBM 650 RAMAC, onde RAMAC significa “Random Access Method of Accounting and Control”[20]. O sistema de discos em si era chamado de IBM 350 e IBM 355, respectivamente para o 305 ou 650.

O conjunto de 50 discos tinha uma capacidade de armazenamento de 5 milhões de palavras de 6 bits, ou seja, aproximadamente 3,75 megabytes. Um dos discos individuais é mostrado na Figura Parte VI: Da Válvula ao Transistor -13.

**Figura Parte VI: Da Válvula ao Transistor -13: Um dos discos do 350.**[21] 1/2

Um IBM 305 com uma unidade de disco 350 era alugado em 1957 por cerca de 3.200 dólares mensais. O preço de aquisição seria algo na ordem de 160 mil dólares, ou seja, 42.600 dólares por megabyte. Hoje (novembro de 2015), um HD externo com 1 terabyte (um milhão de megabytes) custa no Brasil 250 reais, ou seja, um preço de 0,00025 reais por megabyte.

Mais de mil sistemas 350 foram construídos até serem considerados obsoletos em 1962.

## 6.17 Logic Theorist – 1956

No início dos anos 1950, Herbert A. Simon (Estados Unidos, 1916-2001), futuro ganhador do prêmio Nobel de Economia por sua teoria da racionalidade limitada em tomada de decisões, estava prestando consultoria na Rand Corporation (sem relação com a Remington Rand). Durante uma das visitas ele viu uma impressora que estava imprimindo um mapa com caracteres normais e sinais de pontuação. Ao observar isso, ele percebeu o quanto computadores poderiam ser usados para manipular símbolos e não apenas números.

---

pg

[20] Tradução: Método de acesso aleatório para contabilidade e controle.

[21] "RAMAC 305 disk" by I, Deep silence. Licensed under CC BY 2.5 via Commons - https://commons.wikimedia.org/wiki/File:RAMAC_305_disk.JPG#/media/File:RAMAC_305_disk.JPG

O programa que imprimia tal mapa foi escrito por Allen Newell (Estados Unidos, 1927-1992) da Rand, o qual apaixonou-se pela ideia de usar computadores para manipular símbolos após assistir a uma palestra sobre correspondência de padrões.

Newell e Simon passar a trabalhar juntos em 1955 e 1956 examinando a possibilidade de que computadores pudessem pensar. Eles resolveram trabalhar em um programa que fosse capaz de provar os teoremas do livro Principia Mathematica de Bertrand Russel. Eles obtiveram o apoio de John C. Shaw (1922-1991), programador de computadores da Rand.

Eles conseguiram idealizar e escrever um programa para realizar a tarefa de provar os teoremas. Esse programa ficou conhecido como Logic Theorist[22]. A primeira versão do programa foi escrita em um pseudocódigo e era executada de forma simulada por seres humanos. Eles juntaram familiares e alguns estudantes de graduação e cada um recebeu um cartão que descrevia uma das regras do programa. Dessa forma, simulavam a execução do programa de forma mecânica como se fossem um computador. Mais tarde, Shaw conseguiu implementar o programa em um dos computadores da Rand.

O Logic Theorist provou 38 dos 52 problemas apresentados nos primeiros capítulos dos principia; inclusive para um deles o programa produziu uma prova mais elegante do que a de Russel. Eles tentaram publicar essa prova no The Journal of Symbolic Logic, mas o artigo foi rejeitado sob a alegação de que publicar uma nova prova para um teorema conhecido não era originalidade suficiente. Provavelmente os avaliadores não se deram conta de que a prova tinha sido a primeira criação totalmente original produzida por um computador.

Em 1956, John McCarthy, Marvin Minsky, Claude Shanon e Nathan Rochester organizaram a primeira conferência sobre inteligência artificial, termo que até então não existia e que foi criado por McCarthy especialmente para a conferência. Newell e Simon foram à conferência apresentar o Theorist, mas foram recebidos sem nenhum entusiasmo. Aparentemente os pais da inteligência artificial não foram capazes de perceber que naquele momento Newell e Simon já tinham conseguido realizar aquilo que eles próprios ainda buscavam.

O Theorist era um programa que usava uma técnica que até hoje é padrão em sistemas de inteligência artificial, a “busca”. Um programa baseado em busca parte normalmente de um estado inicial, que poderia ser um dos axiomas da lógica, no caso, e a cada instante aplica uma regra de transformação tomada de um conjunto de regras válidas. Por exemplo, o axioma inicial pode ser transformado a partir de alguma regra de transformação lógica como “modus ponens”: Se P->Q é verdadeiro e P é verdadeiro, então Q é verdadeiro. Assim, uma expressão como “P e P->Q” poderia ser transformada em Q, mantendo-se verdadeira.

O problema que eles perceberam imediatamente era que o número de regras que se poderia aplicar era relativamente grande. Ora, partindo de um estado inicial, digamos

[22] Tradução: Teórico lógico.

que apenas 10 regras existam, então as transformações possíveis em 1 passo são 10, em 2 passos são 100 e em 3 passos são 1000. Ou seja, o número de transformações possíveis é de $10^{passos}$. Esse crescimento exponencial logo torna o processo de busca inviável, mesmo para o mais rápido dos computadores. Seria necessário decuplicar a velocidade do computador apenas para poder avançar um único passo a mais na busca no mesmo tempo.

Então eles estabeleceram uma técnica que desde então é usada para que se possa tratar com problemas desse tipo: a heurística. Uma heurística nada mais é do que um critério para eliminar determinados caminhos na busca. Determinadas regras podem até ser possíveis de aplicar, mas você pode ignorá-las ou pelo menos não as priorizar em função de outras regras que são mais promissoras dentro de um dado critério.

A linguagem que Newell, Simon e Shaw criaram para escrever o Logic Theorist, eles chamaram de IPL, Information Processing Language. Era uma linguagem de manipulação de símbolos (não apenas números) baseada em listas e funções, um paradigma de programação que acabou sendo usado depois por John McCarthy para a definição da linguagem LISP.

## 6.18 MUSIC – 1957

Embora o CSIRAC tenha sido o primeiro computador a gerar música e o Ferranti Mark I o primeiro a produzir música que foi gravada, o primeiro programa especificamente projetado para ajudar usuários de computador a produzirem sons musicais foi o MUSIC, criado pelo engenheiro Max Mathews (Estados Unidos, 1926-2011) dos laboratórios Bell em 1957.

O MUSIC foi um primeiro protótipo ou prova de conceito para uma série de outros programas com grande repercussão e uso. O MUSIC foi inicialmente rodado em um computador IBM 704 na Bell.

Todos os programas da série eram baseados em uma biblioteca de rotinas para processamento de sinais e síntese de som, as quais eram invocadas através de códigos de operação (opcodes). Esses códigos podiam ser programados pelo usuário em um arquivo de texto, para produzir a textura de um instrumento, o que definira o tipo de som que seria tocado. Podia ser, por exemplo mais parecido com um violino, flauta ou piano, ou ainda completamente original. Um segundo arquivo definia as notas que seriam tocadas: duração e tonalidade, além de outros parâmetros. Assim, a partir de 1957 já era possível que um usuário escrevesse música para ser tocada no computador usando uma linguagem de propósito específico que não era a linguagem de máquina.

A primeira peça tocada pelo IBM 704 foi composta por um colega de Max, Newman Guttman, e chamava-se “The Silver Scale”. Ela pode ser ouvida a partir de http://opinionator.blogs.nytimes.com/2011/06/08/the-first-computer-musician/.

Consta que em 1961, Max produziu outra façanha, ao fazer uma máquina sintetizar voz humana cantando a música “Bicycle Build for Two”, que também pode ser ouvida no site mencionado anteriormente. Dizem que Stanley Kubrick ficou tão

impressionado ao ouvir essa voz sintetizada cantando que resolveu usar parte da melodia, que acabou ficando conhecida como “Daisy, Daisy” na sequência em que a memória do Computador HAL 9000 começa a perder a consciência ao ser lentamente desligado no filme “2001: Uma odisseia no espaço”. O computador HAL 9000 cantaria então uma música que ele teria aprendido na “infância”. Uma outra curiosidade bem conhecida sobre este computador também é que “HAL” é uma codificação criptográfica ao estilo de Al-Kindi para “IBM” na qual se substitui cada letra pela imediatamente anterior no alfabeto.

## 6.19 Scanner - 1957

Mais ou menos na mesma época em que os computadores aprendiam a cantar eles também aprendiam a ver. A invenção do scanner é reputada a Russel Kirsch (Estados Unidos, 1929), que trabalhava em 1957 com o SEAC, ou Standards, Electronic Automatic Computer, o primeiro computador programável usado pelo governo federal Norte-Americano para fins civis.

O scanner de computador pode ser considerado um descendente das máquinas fotográficas e aparelhos de fac-símile (fax), bem como das telefotos; tudo tecnologia desenvolvida a partir do Século XIX. Mas a novidade do scanner construído por Kirsch e sua equipe estava no fato de que pela primeira vez na história, um computador guardou na memória a representação binária de uma imagem, ou seja, uma foto digital.

A primeira imagem digitalizada da história é mostrada na Figura Parte VI: Da Válvula ao Transistor -14, e trata-se da digitalização de uma foto de um dos filhos de Kirsch, na época com 3 meses. A foto foi digitalizada em 176 por 176 bits e como cada bit pode representar apenas 0 ou 1, ou seja, preto ou branco, isso impossibilitava tons de cinza.

**Figura Parte VI: Da Válvula ao Transistor -14: A primeira imagem digital.**[23] **1/3**

[23] "NBSFirstScanImage". Licensed under Public Domain via Commons - https://commons.wikimedia.org/wiki/File:NBSFirstScanImage.jpg#/media/File:NBSFirstScanImage.jpg

Porém, a ideia de digitalizar imagens usando apenas bits com 0s ou 1s não foi uma boa escolha. Quando os engenheiros reproduziram a imagem digitalizada em um tubo de raios catódicos, perceberam esse erro. Assim, eles passaram a realizar várias passagens do scanner sobre a imagem com diferentes níveis de sensibilidade à luz. Dessa forma puderam construir uma imagem muito mais realista usando tons de cinza ao invés de preto e branco.

## 6.20 FORTRAN – 1957

Apesar de Zuse ter concebido a primeira linguagem de programação de alto nível em 1946, ela nunca foi implementada até a década de 1970. A primeira linguagem de programação científica de alto nível implementada foi, de fato, FORTRAN, um acrônimo para FORmula TRANslating.

FORTRAN foi concebida por John Backus (Estados Unidos, 1924-2007), segundo ele, por preguiça de ter que escrever tantos comandos para realizar as operações mais simples no computador. De fato, com FORTRAN a quantidade de comandos que precisava ser programada para atingir o mesmo objetivo era cerca de 20 vezes menor.

Backus começou a programar em 1950 no IBM SSEC. No final de 1953 ele escreveu um memorando ao seu chefe com a proposta de desenvolver uma linguagem de programação para o novo IBM 701 que facilitaria o trabalho de programação.

A proposta foi aprovada e ele ganhou inclusive uma equipe de 4 pessoas para ajudá-lo. Em 1954, com a disponibilidade do IBM 704, Backus passou a focar nesta máquina.

Ainda em 1954 ele e sua equipe lançaram um documento onde especificavam a linguagem cujo compilador (na época chamado de “tradutor”) deveria ser lançado em cerca de 6 meses. Porém, o trabalho demorou um pouco mais que o esperado, e em 1957 o primeiro compilador de FORTRAN, ainda cheio de bugs era disponibilizado ao mundo.

O primeiro compilador FORTRAN consistia de 25 mil linhas de código (de máquina). Toda máquina IBM 704 vinha com uma fita magnética contendo o FORTRAN e um manual de 51 páginas (Figura Parte VI: Da Válvula ao Transistor -15).

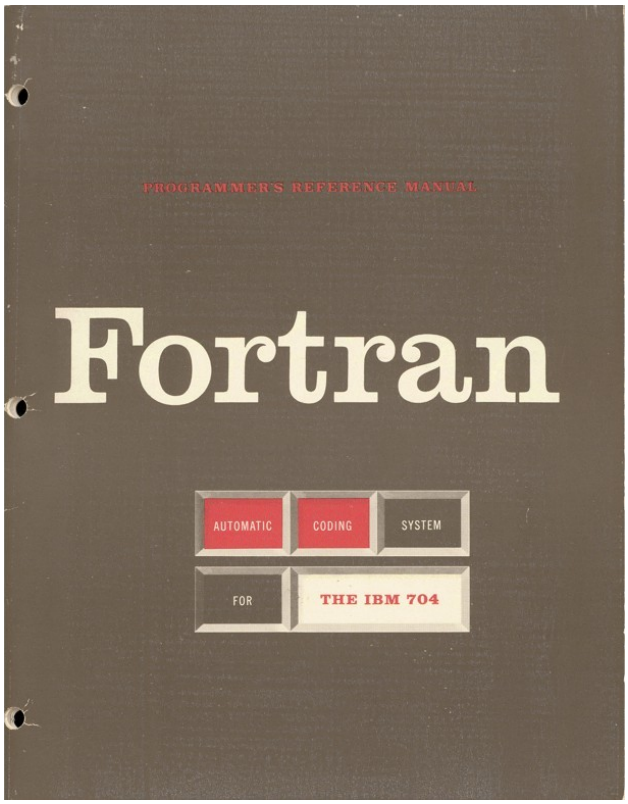


**Figura Parte VI: Da Válvula ao Transistor -15: Primeiro manual de Fortran.**[24] 1/2

Na proposta de 1954, Backus já previa que a linguagem tornaria o processo de programação muito mais rápido. O maior risco e o maior fator de rejeição a um possível compilador de linguagem de alto nível vinham do fato de que muitos programadores duvidavam que um computador pudesse gerar código de máquina através do processo de compilação que fosse mais eficiente do que um código de máquina produzido por um humano.

A realidade, entretanto, mostrava que os humanos tinham grande dificuldade de produzir código de máquina de boa qualidade e, assim, de fato, a máquina se saia muito melhor, produzindo programas em código de máquina mais eficientes do que os humanos seriam capazes de fazer.

Algumas características da proposta inicial do FORTRAN:

- Números inteiros, como 56, +900 e -167, que seriam armazenados em variáveis com um ou dois caracteres alfanuméricos, iniciados pelas letras i, j, k, l, m, ou n. Por isso até hoje muitos programadores fazem suas variáveis de índice (inteiras) serem i, j, k, etc.
- Números de ponto flutuante, como .02, 12.45 e -9.4, que seriam armazenados em variáveis com um ou dois caracteres alfanuméricos iniciados por qualquer letra exceto i, j, k, l, m ou n.

---

[24] "Fortran acs cover" by original uploader was en:User:Muhandis - en:File:Fortran acs cover.jpeg. Licensed under Public Domain via Commons - https://commons.wikimedia.org/wiki/File:Fortran_acs_cover.jpeg#/media/File:Fortran_acs_cover.jpeg

- Operadores binários: +, -, ×, / e ×× (exponenciação).
- Funções predefinidas e definíveis pelo usuário, como sin(*x*) - seno, sqrt(*x*) – raiz quadrada, factl(*x*) – fatorial, max(*a*, *b*, *c*, *d*, *e*) - o maior dentre *a*, *b*, *c*, *d* e *e*.
- Expressões combinando constantes, variáveis e operadores, desde que tivessem no máximo 750 caracteres.
- Uso de parênteses aninhados em expressões complexas. Por exemplo: ((*a*+*b*)**c*+*d*)/*e*.
- Variáveis subscritas, hoje chamadas de vetores ou arrays. Por exemplo A(*n*) significava $A_n$.
- Declaração de arrays ou matrizes pelo comando DIMENSION. Por exemplo, DIMENSION *v*[2,3] definia uma matriz *v* com duas linhas por 3 colunas, as quais podiam ser referenciadas por variáveis subscritas como *v*(1,1), *v*(1,2), etc.
- Números de fórmulas, hoje chamados de rótulos ou números de linha. Um comando em FORTRAN poderia opcionalmente ter um número antes dele para ser referenciado por comandos de repetição ou decisão. Por exemplo, em 45 *x*=*a*+*b*, aplica-se o rótulo 45 ao comando *x*=*a*+*b* (atribuir o valor de *a*+*b* à variável *x*). Assim, esse comando passa a ser referenciado como o comando 45.
- Estruturas de repetição, ou DO (para-faça). O formato originalmente proposto seria algo do tipo DO 10, 14, 50 *i*=4, 20, 2 significando o seguinte: repita os comandos desde o número 10 até o número 14 um certo número de vezes. Depois disso vá para o comando 50. O número de vezes é governado pela variável de loop *i*, que vai valer 4 da primeira vez e depois ter seu valor somado a 2, cada vez que os comandos forem repetidos. A última repetição vai ocorrer quando *i* valer 20. Assim, *i* vai assumir os valores 4, 6, 8, 10, 12, 14, 16, 18, 20 e a repetição, no caso, será feita 9 vezes.
- Estrutura de decisão (IF). Embora as versões mais antigas de FORTRAN usassem IF numérico (decidir para onde ir dependendo do sinal de uma variável), a proposta de 1954 já previa o IF lógico, com instruções como: IF (*a*>=*b*) 23, 56, ou seja, se *a* for maior ou igual a *b* vá para o comando 23. Senão vá para o comando 56.
- GOTO. Um comando hoje em desuso em função dos princípios da programação estruturada, mas que fazia o programa ir diretamente para a instrução cujo número era passado logo depois. Por exemplo, GOTO 56 ia diretamente para o comando 56.
- STOP, que fazia o computador parar.
- Instruções de entrada e saída: *read*, *punch*, *print*, *read tape*, *write tape*, *end file*, *rewind*, *backspace*, *read drum* e *write drum*.

FORTRAN foi durante mais de 30 anos a principal linguagem de programação científica e ainda hoje é usada em muitas áreas onde se necessita de alta precisão e eficiência.

Uma característica de FORTRAN que Backus já visualizava em 1954 era o fato de que as linguagens de máquina eram diferentes de uma máquina para outra, mas um programa escrito em FORTRAN poderia rodar em qualquer máquina desde que compiladores específicos fossem construídos. Assim, quando um usuário trocasse um modelo de computador por outro, não precisaria mais reescrever todos os seus programas. Uma vantagem enorme, hoje conhecida como “portabilidade”.

O problema é que muitos compiladores FORTRAN levemente diferentes foram feitos, os quais eram, portanto, incompatíveis. Isso levou a associação americana de padrões a estabelecer um padrão para a linguagem FORTRAN em 1966, o qual ficou conhecido como FORTRAN IV. Esse padrão foi revisado em 1977 com o lançamento do FORTRAN 77 e por último em 1990, com o FORTRAN 90.

## 6.21 ALGOL – 1958

Nos anos de 1951 a 1954 algumas tentativas de desenvolver linguagens de alto nível, também chamadas de linguagens algorítmicas foram feitas na Europa, mas sem avançar muito logo de início. Porém, essas tentativas frustradas prepararam os espíritos para a chegada de FORTRAN que, embora ainda não implementado, já tinha seu primeiro relatório amplamente divulgado desde 1954.

Ao mesmo tempo em que a indústria, através da IBM e da Bell trabalhava para criar uma linguagem de alto nível para aplicações científicas, a academia não estava parada. De fato, já em 1947 tinha sido fundada nos Estados Unidos a ACM – Association for Computing Machinery, a qual organizava regularmente encontros e simpósios.

Na Europa, um simpósio internacional sobre computação automática foi organizado em 1955 na Alemanha, em Darmstadt. “Computação automática” ou “programação automática” era o nome que se dava à programação em alto nível naquela época.

Vários participantes dessa conferência concordaram que era necessário definir uma linguagem algorítmica, independente de máquina, universal e única para ser usada por todos, ao invés de ter várias linguagens diferentes.

Assim, na mesma conferência foi criado o comitê GAMM (uma sigla em alemão que significa Sociedade para Matemática e Mecânica Aplicadas) para estudar esse assunto e apresentar propostas.

Porém, em 1957 quando quase completavam seu trabalho na especificação de uma nova linguagem algorítmica, os membros do GAMM se deram conta de que ela seria apenas mais uma em meio a outras. Assim, eles teriam é que se esforçar para unificar as diferentes vertentes. Buscando um maior diálogo com a América, convidaram a ACM para realizar uma conferência conjunta com a finalidade de discutir a proposta de uma linguagem universal.

Ao compararem a proposta do GAMM com propostas elaboradas pela ACM, eles acharam muitos pontos em comum e assim, a conferência para unificação das propostas foi realizada em Zurique em meados de 1958. Estavam presentes quatro

membros da ACM e quatro do GAMM. Os princípios para a elaboração da linguagem universal seriam os seguintes:

- Ela seria a mais parecida possível com a linguagem matemática, de forma que pudesse ser lida por matemáticos com o mínimo esforço de interpretação.
- Deveria ser possível usar a linguagem para descrever processos numéricos em publicações.
- A nova linguagem teria que ser imediatamente traduzível para código de máquina pela própria máquina.

A conferência de Zurique produziu assim o relatório do ALGOL 58. Como havia pouquíssima padronização em relação a comandos de entrada e saída de dados, eles decidiram deixar de lado este aspecto da computação e se concentrar apenas nas questões mais abstratas, o que depois revelou ser um erro, pois foi um dos fatores que fizeram com que ALGOL, a Linguagem Algorítmica (ALGOrithmic Language) que eles definiram, não chegasse a ser tão popular quanto FORTRAN ou COBOL na ciência e no comércio.

O primeiro compilador ALGOL foi criado ainda em 1958 por Friedrich L. Bauer (Alemanha, 1924-2015) para rodar no computador Z22, um computador a válvulas, o sétimo modelo construído por Konrad Zuse.

Várias propostas de melhorias foram publicadas na revista Communications of the ACM em 1959 (volume 2, número 7). Esse número da revista também ficou conhecido como “Boletim ALGOL”.

Em 1959 o GAMM reuniu 50 membros em Paris para selecionar 7 para o comitê final de ALGOL. Nos EUA a ACM também realizou um encontro para selecionar 7 outros delegados. Finalmente, em janeiro de 1960 ocorre em Paris a conferência que produziu o relatório ALGOL 60. Entre outras pessoas, estiveram na conferência John Backus (criador do FORTRAN), Peter Naur e John McCarthy.

Em 1960 também ocorreu a fundação da IFIP (International Federation for Information Processing) uma federação de sociedades científicas de vários países do mundo, da qual a ACM tornou-se membro. Em 1962 a manutenção e desenvolvimento do ALGOL foi assumido pela IFIP, em especial, pelo grupo de trabalho WG 2.1 (Linguagens Algorítmicas e Cálculo).

Uma das diferenças de ALGOL em relação a FORTRAN está no fato de que em ALGOL as variáveis devem ter seu tipo declarado. Em FORTRAN é a primeira letra do nome da variável que define o tipo dela. Assim, por exemplo, uma variável iniciada com *i* é sempre inteira e uma variável iniciada por *x* é sempre real. Em ALGOL seria necessário declarar, antes de usar a variável se ela era inteira ou real e sua letra inicial podia ser qualquer uma. Isso parece desvantajoso à primeira vista, mas tem seu valor, porque com ALGOL, por exemplo uma variável que representasse um contador poderia ser declarada como “*integer contador*”, em FORTRAN ela teria que ter seu nome iniciado com uma letra entre *i* e *m* para ser inteira, como por exemplo “*icontador*”, o que fica estranho.

Além disso, o fato de se poder declarar o tipo da variável também abriu caminho para que as linguagens pudessem permitir que o programador criasse seus próprios tipos personalizados. Assim, em vez de ter apenas inteiro e reais, você poder booleanos, textos, ou mesmo temperaturas, CPF, data, ISBN, etc.

Segue abaixo um exemplo de programa em ALGOL:

```
proc abs max = ([,]real a, ref real y int i, k)real:
comment O maior elemento absoluto da matriz a, de tamanho ⌈a por 2⌈a é armazenado em y e os índices
deste elemento em i e k; comment
begin
  real y:=0; i:=⌊a; k:=2⌊a;
    for p from ⌊a to ⌈a do
      for q from 2⌊a to 2⌈a do
        if abs a[p,q] > y then
          y:=abs a[p,q];
          i:=p; k:=q
        fi
      od
    od;
    y
end # abs max #
```

Para descrever a estrutura da linguagem ALGOL, John Backus criou uma notação que depois foi aperfeiçoada por Peter Naur e ficou conhecida como Notação Backus-Naur (BNF, de “Backus-Naur Form”). Até hoje ela é a notação padrão para especificação de linguagens de programação.

ALGOL praticamente não é mais usada, mas ela deixou influências em muitas outras linguagens correntes de programação como C, Pascal, Java e Python.

## 6.22 Tênis para Dois – 1958

O primeiro jogo eletrônico para computador com animação em tela foi criado em 1958 por William Higinbotham (Estados Unidos, 1910-1994). Tratava-se de uma simulação de jogo de tênis, que ficou conhecida como “Tennis for two”, ou “Tênis para dois”.

Consta que Higinbotham resolveu criar o jogo para reduzir a monotonia que sentiam os visitantes do Laboratório Nacional Brookhaven, onde ele trabalhava.

Ele usou um computador analógico Donner Model 30 ligado a um osciloscópio, que é um instrumento com uma tela de vídeo CRT. Com este osciloscópio, ele desenhava na tela um jogo de tênis visto de perfil (Figura Parte VI: Da Válvula ao Transistor -16) com uma rede, o chão e uma “bolinha” representada por um ponto que ficava pulando de um lado para o outro da tela, deixando um pequeno rastro atrás de si.


**Figura Parte VI: Da Válvula ao Transistor -16: Tênis para dois.**[25] 1/2

Ele conta que teve a ideia quando soube que o computador era capaz de calcular a trajetória de projéteis. Daí ele usou os circuitos que faziam estes cálculos para mover a “bolinha” na tela do osciloscópio. O circuito podia sentir quando a bolinha tocava o chão, para inverter seu movimento, e também se ela tocasse a rede. Os jogadores usavam um botão de girar (potenciômetro) para definir o ângulo da raquete (que não aparecia na tela) e um botão de pressionar para indicar o momento da raquetada. Ao ser atingida a bolinha, a máquina emitia um bip.

Consta que o projeto levou apenas quatro horas para ser desenhado e cerca de duas semanas para ser construído.

Como as outras exibições do laboratório eram basicamente estáticas, o Tênis para dois rapidamente tornou-se a vedete das exposições, e centenas de pessoas faziam fila para jogar.

Porém, após algumas poucas exibições o jogo foi desmantelado e ficou esquecido por mais de 20 anos até que seu autor foi convocado a depor como testemunha em um processo judicial entre duas produtoras de videogames, a Atari e a Odyssey. A partir dessa redescoberta, ele foi reconhecido como o avô dos videogames.

## 6.23 LISP – 1958

FORTRAN é a mais antiga linguagem e programação ainda em uso. A segunda mais antiga ainda em uso é uma linguagem que foi criada especificamente para manipulação de símbolos e não de números. Essa linguagem chama-se LISP (LISt Processing) e foi criada por John McCarthy, personagem que já foi citado algumas vezes nas seções anteriores.

---

[25] "Tennis For Two on a DuMont Lab Oscilloscope Type 304-A" by Brookhaven National Laboratory - Screenshot. Licenced under Public Domain via Wikimedia Commons - https://commons.wikimedia.org/wiki/File:Tennis_For_Two_on_a_DuMont_Lab_Oscilloscope_Type_304-A.jpg#/media/File:Tennis_For_Two_on_a_DuMont_Lab_Oscilloscope_Type_304-A.jpg

Lembram-se de quando falamos na Tese Church-Turing? Turing criou uma definição de computabilidade baseada em sua Máquina A, e Church também criou uma baseada em um princípio bem diferente, chamado Cálculo λ (Lambda). O Cálculo Lambda é uma linguagem baseada em substituições de símbolos. Ao invés de simplesmente operar numericamente com variáveis como usualmente fazemos na aritmética, com o Cálculo Lambda trabalha-se diretamente com os símbolos. Por exemplo, pode-se escrever uma expressão que define que $a(x+y)$ é equivalente a $ax+ay$. Fazendo substituições de símbolos, assim, a partir de suas definições, o cálculo lambda serve como um mecanismo de simplificação, que vai reduzindo as expressões até um formato que é considerado final: a saída do programa.

Pois bem, inspirado na notação de Church e também na linguagem IPL usada por Newell, Simon e Shaw para implementar o Logic Theorist, McCarthy, nessa época trabalhando no MIT, escreveu um artigo chamado "Recursive Functions of Symbolic Expressions and their Computation by Machine, Part I"[26]. Esse artigo, escrito em 1958 foi publicado na revista "Communications of the ACM" em 1960. Já a parte II, se é que foi escrita, nunca foi publicada. McCarthy mostrou que era possível construir uma linguagem baseada em transformações sobre listas e funções e que essa linguagem simbólica seria Turing-completa.

O compilador LISP foi implementado pela primeira vez em um IBM 704 por Stephen Russel (Estados Unidos, 1937), o qual também ficou famoso anos depois pela implementação do videogame Spacewar!. Mas essa primeira implementação do LISP era apenas uma prova de conceito. //ATENÇÃO REVISÃO: O NOME DO VIDEOGAME É 'SPACEWAR!' COM O PONTO DE EXCLAMAÇÃO. NÃO SE TRATA DE ERRO DE PONTUAÇÃO AQUI//

A primeira implementação completa de LISP foi feita em 1962 por Tim Hart e Mike Levin no MIT. Desde sua criação, LISP foi uma linguagem fortemente utilizada pela comunidade de Inteligência Artificial. Vários sistemas de Inteligência artificial (IA) foram implementados em LISP.

LISP foi a primeira linguagem implementada a permitir o uso de "recursão", um conceito que não era permitido pelas primeiras implementações de FORTRAN. Uma função é recursiva se ela for definida em termos dela própria. Por exemplo, a função "fatorial" pode ser definida de forma iterativa como $n! = n\times(n-1)\times(n-2)\times...\times2\times1$. Assim, $5! = 5\times4\times3\times2\times1$. Mas a função fatorial também pode ser definida de forma recursiva como $n! = n\times(n-1)!$. Assim, $5! = 5\times4!$. Porém, para que a recursão não continue indefinidamente com a função sempre chamando a si própria é necessário estabelecer que a expressão acima só se aplica se $n$ for maior do que 0. Define-se, adicionalmente, então, que o fatorial de 0 é 1: $0! = 1$.

FORTRAN não permitia que o nome de uma função fosse usado em sua definição; assim, funções definidas recursivamente eram impossíveis. Mas LISP permitia. O

[26] Tradução: Funções Recursivas de Expressões Simbólicas e sua Computação por Máquina, Parte I.

programa abaixo, escrito em uma versão moderna do LISP é uma definição recursiva da função fatorial:

```
DEFINE ((
  (FATORIAL (LAMBDA (X)
    (COND ((EQUAL X 0) 1)
      (T (TIMES X (FACTORIAL (SUB1 X)))))))
))
```

A quantidade de parênteses usada em programas LISP frequentemente fazia os programadores referirem-se aos seus programas como “selvas de parênteses”. Esse uso intensivo de parênteses é devido ao fato de que todas as expressões da linguagem serem, a princípio, expressões baseadas em listas. Se você quiser somar $x+y+z$ em LISP teria que escrever (SUM X Y Z). Porém, versões mais modernas da linguagem acabaram permitindo que a notação usual $x+y+z$ fosse usada em muitas situações.

## 6.24 Até Aqui...

Vimos que a década de 1950 testemunhou o início da computação comercial acessível às empresas e demais organizações. Nessa década surgiram os sistemas operacionais, o disco magnético e os primeiros computadores transistorizados.

Três importantes linguagens de programação surgiram nos anos 1950: FORTRAN, ALGOL e LISP. Duas delas ainda são bastante usadas e a outra (ALGOL) deixou vários descendentes que hoje são dominantes na indústria. Na área comercial, a década viu também a evolução do “compilador” A-0 que gerou o FLOW-MATIC, o qual acabou sendo a principal influência para a definição da linguagem padrão do comércio, o COBOL, logo no início dos anos 1960.

Durante os anos 1950 viu-se também que era possível usar computadores para produzir música e reconhecer imagens. A inteligência artificial, que começou a ser imaginada como possibilidade nos anos 1940 vê surgir seu primeiro triunfo: o Logic Theorist.

Em termos de equipamentos, os principais fornecedores dessa década foram a IBM, com seus modelos 701, 650 e 704 e a Remington Rand com o UNIVAC. Em termos de computador pessoal, merece menção o Simon que com um processador de apenas 2 bits foi o primeiro equipamento de computação acessível ao cidadão comum.

Toda essa evolução, mas especialmente a evolução do transistor, das memórias de núcleo magnético, dos sistemas operacionais e da computação de tempo real preparou o caminho para desenvolvimentos ainda maiores na década seguinte, cuja marca seria o surgimento do computador pessoal.