Burroughs MCP

Maszyna zaprojektowana pod ALGOL, ze stosem wywiedzionym z notacji polskiego logika i z systemem operacyjnym pisanym w języku wysokiego poziomu — której potomek w 2026 roku pracuje na procesorach Intela, w VMware i w chmurze, choć jej legenda obiecuje więcej, niż potwierdzają źródła.

To jest historia Burroughsa B5000 i jego systemu MCP (Master Control Program): od firmy, która przez siedemdziesiąt lat produkowała maszyny sumujące, przez zespół z Pasadeny, który na przełomie lat pięćdziesiątych i sześćdziesiątych postanowił zbudować komputer dla języka ALGOL, po Unisysa i jego linię ClearPath MCP, sprzedawaną dziś jako oprogramowanie na zwykłe serwery x86. Po drodze opowiem, jak działały stos, deskryptory i znaczniki w pamięci, dlaczego „pierwsza pamięć wirtualna” to spór z Manchesterem, w jakim języku napisano pierwszy MCP i czym się skończył test penetracyjny z 1981 roku. Na końcu — co z tego zostało w 2026 roku i jak to uruchomić we własnej przeglądarce.

> Umiejscowienie w serii: to jest strona o **Burroughs MCP** — od maszyn sumujących Burroughsa i biuletynu „The Descriptor” z lutego 1961 roku po ClearPath MCP 20.0 i mapę drogową do wydania 23.0. Należy do księgi „Zanim był pecet”. Świat IBM, z którym Burroughs konkurował i od którego świadomie odszedł, opisuje strona **Mainframe i z/OS**. Segmentowaną pamięć, którą B5000 wprowadził do handlu, rozwinął dalej **Multics**. Najbliższym krewnym ideowym jest **IBM i: System/38 i AS/400** — tam także znaczniki w pamięci okazują się akceleratorem, a nie strażnikiem, i tam także system przeżył własny procesor. Rówieśnikami z innych firm są **VMS i OpenVMS**, **Tandem NonStop** i **HP MPE**. Emulator B5500 w przeglądarce należy do tej samej rodziny narzędzi, co opisane na stronie **Retro w praktyce 2026**.

□□ Skróty użyte na tej stronie (85)

Skrót	Co znaczy
ACM	Association for Computing Machinery: stowarzyszenie informatyków
AFIPS	American Federation of Information Processing Societies: federacja, współorganizator konferencji z 1985 roku
ALGOL	Algorithmic Language: język programowania, dla którego zbudowano B5000
APL	A Programming Language: język programowania, dostępny na B6700
AS/400	linia serwerów IBM z 1988 roku, bohater sąsiedniej strony
ASCII	standardowy kod znaków; tu pliki tekstowe z taśmami Mark XIII

Skrót	Co znaczy
ATC	Air Training Command: dowództwo szkolenia lotnictwa USA, gdzie Kay poznał B220 i B5000
AWS	Amazon Web Services: chmura publiczna Amazona
BASIC	prosty język programowania
BCD	Binary-Coded Decimal: zapis cyfr dziesiętnych po cztery bity
BCP	Basic Control Program: prostszy system dla małych konfiguracji Medium Systems
BNA	Burroughs Network Architecture: architektura sieciowa Burroughsa
BOS, SL	dolna i górna granica stosu zadania w B6500
CANDE	Command and Edit: system obsługi terminali i edycji w MCP
CBI	Charles Babbage Institute: archiwum historii informatyki na Uniwersytecie Minnesoty
CC	Creative Commons: rodzina wolnych licencji (BY — z podaniem autora, SA — na tych samych warunkach, NC — tylko użytek niekomercyjny, ND — bez utworów zależnych)
CC0	Creative Commons Zero: zrzeczenie się praw, czyli oddanie utworu do domeny publicznej
CMOS	technologia układów scalonych; tu własne procesory Unisysa do 2014 roku
CMP	architektura serwerów Unisysa budowanych z płyt komórkowych (cell boards)
COBOL	Common Business-Oriented Language: język aplikacji biznesowych
CSS	ClearPath MCP Software Series: programowa wersja MCP
CUBE	biblioteka programów użytkowników B5500
DD, SD	Data Descriptor, Segment Descriptor: deskryptory danych i segmentów w B6500
DEC	Digital Equipment Corporation: producent minikomputerów i systemu VMS
DF MCP	Disk File MCP: dyskowy MCP dla B5500, pisany w ESPOL
DMS, DMS II	system zarządzania danymi dużych systemów Burroughsa
DPS 8	linia komputerów Honeywella, konkurent A Series
EBCDIC	kod znaków IBM, dostępny też w Medium Systems
ECC	Error Correcting Code: kod korekcji błędów pamięci
ECS	segment Unisysa, do którego należy ClearPath Forward
EMC	Dell EMC: dostawca pamięci masowych
ESPOL	Executive Systems Problem Oriented Language: odmiana ALGOL-u, w której pisano MCP od B5500
FJCC	Fall Joint Computer Conference: konferencja, na której opisano B1700 (1972)
FLEX	maszyna Alana Kaya, w której referencje obiektów uogólniały deskryptory B5000
FORTRAN	Formula Translation: język obliczeń naukowych

Skrót	Co znaczy
FORTRANSIT	wczesny kompilator FORTRAN-u IBM
GEMCOS	składnik oprogramowania B 5900 i A Series wymieniany przez Datapro
HLL	high-level language: język wysokiego poziomu
HMP	Heterogeneous Multiprocessing: linia ClearPath z 1996 roku
HP	Hewlett-Packard; tu w nazwie strony HP MPE
IBM	International Business Machines
IEEE	Institute of Electrical and Electronics Engineers: autor tablicy Milestone dla Atlasa
IPsec, IKEv2	protokoły szyfrowania i wymiany kluczy w sieciach IP
KB, MB, GB	kilobajt, megabajt, gigabajt
L&S	License and Support: przychody Unisysa z licencji i wsparcia
LINC	oprogramowanie A Series wymieniane przez Datapro
LSI	Large-Scale Integration: układy scalone wielkiej skali integracji
MCP	Master Control Program: system operacyjny Burroughsa, dziś ClearPath MCP
MCP/AS	nazwa MCP w latach 90.
MCPvm	wariant MCP działający jako maszyna wirtualna na procesorach Intela (2002)
MIPS	Million Instructions Per Second: przybliżona miara mocy procesora
MIT	tu: licencja MIT, liberalna licencja oprogramowania
MMCP	Micro Master Control Processor: mikrokodowany procesor B 5930
MOS	technologia tranzystorów; tu pamięć półprzewodnikowa B 5930
MPE	system operacyjny minikomputerów HP, bohater sąsiedniej strony
MQ	WebSphere MQ: kolejka komunikatów IBM
MSCW	Mark Stack Control Word: słowo zaczynające rekord aktywacji procedury na stosie B6500
NASA	National Aeronautics and Space Administration: drugi klient B5000
NDL II	składnik oprogramowania B 5900 i A Series wymieniany przez Datapro
NEWP	język, w którym od początku lat 80. pisany jest MCP
NMAH	National Museum of American History: muzeum Smithsonian
NT	Windows NT: system Microsoftu
NTBS	New Technology Business Solutions: system bankowy Co-operative Bank
OH	Oral History: seria wywiadów historii mówionej w Charles Babbage Institute
OOP	object-oriented programming: programowanie obiektowe
OS 2200	system linii Sperry/UNIVAC w Unisysie, drugi obok MCP

Skrót	Co znaczy
OSIL	asembler na Burroughs 220, w którym napisano pierwszy MCP
PL/I	język programowania IBM, dostępny na B6700
PRT	Program Reference Table: tablica odniesień programu w B5000
RCA	Radio Corporation of America: niedoszły partner Burroughsa
RPG	Report Program Generator: język aplikacji biznesowych
SDL	Software Development Language: język, w którym napisano MCP II dla B1700
SJCC	Spring Joint Computer Conference: konferencja, na której opisano stos B6500 (1968)
SMB	protokół udostępniania plików Windows
SPO	stanowisko operatora B5500, opisane w wiki emulatora
TLS	Transport Layer Security: szyfrowanie połączeń sieciowych
TSS	Time-Sharing System: system podziału czasu B5500/B5700
UNIVAC	linia komputerów Sperry Rand
USA	Stany Zjednoczone
USASCII	wczesna nazwa kodu ASCII
UTC	pierwszy klient B5000; rozwinięcie niepotwierdzone
VMS	system operacyjny DEC, dziś OpenVMS
WFL	Work Flow Language: język sterowania zadaniami w MCP
WJCC	Western Joint Computer Conference: konferencja, na której Barton ogłosił artykuł (1961)
x86	rodzina procesorów Intela i zgodnych

□□ Leksykon: nazwy i pojęcia z tej strony (29)

Nazwa	Co to jest
A Series	linia dużych systemów Burroughsa i Unisysa od 1984 roku
Atlas	komputer z Manchesteru i Ferranti, z pamięcią wirtualną ze stronicowaniem (1962)
B1700	mikroprogramowana maszyna z rodziny Small Systems (1972)
B5000	duży komputer Burroughsa, opisany w 1961, dostarczany od 1963 roku
B5500	B5000 z dyskiem i dyskowym MCP, ogłoszony w 1964 roku
B6500	następca linii 5000, z trzybitowymi znacznikami i drzewem stosów
bit flagi	jedyny bit znacznika w słowie B5000
bit obecności	bit deskryptora mówiący, czy segment jest w pamięci głównej

Nazwa	Co to jest
bud	„pączkowanie” zadania B6500 w dwa niezależne
ClearPath	linia serwerów i systemów Unisysa od 1996 roku, z MCP i OS 2200
deskryptor	słowo opisujące segment pamięci: adres, rozmiar, obecność
Developer Studio	płatny pakiet Unisysa do tworzenia aplikacji MCP (2024)
Display	rejestry B6500 wskazujące procedury będące w zasięgu
Extended ALGOL	rozszerzony ALGOL Burroughsa, podstawa ESPOL
Libra	seria serwerów ClearPath z MCP
Mark XIII	wydanie oprogramowania B5500 z 1971 roku, dostępne w emulatorze
MCP Express	bezpłatna wersja MCP na pecet z roczną licencją (2016)
Medium Systems	dziesiętna rodzina Burroughsa B2500–B4900
notacja polska	zapis wyrażeń bez nawiasów, wymyślony przez Jana Łukasiewicza
retro-B5500	emulator B5500 w przeglądarce (Williams, Kimpel, 2013)
S-maszyna	interpreter języka w mikrokodzie B1700
Saguaro Stack System	drzewo stosów B6500, dziś „stos kaktusowy”
Software Series	wersja ClearPath MCP wyłącznie programowa
stos	pamięć tymczasowa, z której operacje zdejmują argumenty i na którą kładą wyniki
Stream Procedures	procedury trybu znakowego w ALGOL-u B5000
sylaba	12-bitowa (w B6500 8-bitowa) jednostka rozkazu
Unisys	firma powstała w 1986 roku z Burroughsa i Sperry
zejścia rekurencyjne	metoda kompilacji, w której regułom gramatyki odpowiadają procedury
znacznik	bity słowa określające jego typ; w B6500 trzy

Zastrzeżenie zakresu. To nie jest podręcznik operatora MCP ani katalog modeli Burroughsa i Unisysa. Najwięcej miejsca dostaje to, co obiegowa opowieść upraszcza: kto zaprojektował B5000, czy rzeczywiście nie miał asemblera, czy był pierwszy z pamięcią wirtualną i czy znaczniki w pamięci czyniły go odpornym na włamania. Wiele najbarwniejszych szczegółów pochodzi ze wspomnień uczestników — z konferencji zorganizowanej w 1985 roku, ponad dwadzieścia lat po fakcie, i z tekstu Richarda Waychoffa spisanego około 1979 roku. To są relacje ludzi, którzy przy tym byli, ale nie dokumenty firmowe; gdzie liczba pochodzi tylko ze wspomnień, mówię to wprost. Ceny podaję w dolarach z epoki, bez przeliczania na dzisiejsze.

Wprowadzenie: legenda o maszynie, która wyprzedziła swój czas

Legenda w jednym akapicie

Kto czytał cokolwiek o historii architektury komputerów, zna tę opowieść w jakiejś wersji: *Robert Barton w 1961 roku zaprojektował B5000 — maszynę stosową bez asemblera, której system MCP jako pierwszy na świecie napisano w języku wysokiego poziomu. Była to pierwsza maszyna z pamięcią wirtualną, a znaczniki w pamięci czyniły ją odporną na przepełnienia bufora. Linia przetrwała do dziś jako Unisys ClearPath, emulowana na Xeonach, a nazwę MCP zapożyczył film „Tron”.*

W jej sercu jest prawda — A. J. W. Mayer w 1982 roku zatytułował artykuł o B5000 pytaniem, czy dwadzieścia lat później wciąż wyprzedza czasy. Ale prawie każdy człon tej opowieści, przyłożony do źródeł, okazuje się uproszczeniem, prawdą z zastrzeżeniem albo czymś, czego nie da się ustalić.

Rozbiór człon po członie

Człon legendy	Werdykt	Dlaczego
„Barton zaprojektował B5000”	prawda częściowa	Był głównym pomysłodawcą, ale artykuł z maja 1961 roku podpisał jako konsultant z Altadeny, a sam nazywał się „sabotażystą przemysłowym”. Maszynę zbudował zespół z Pasadeny: Lloyd Turner, David Dahm, Paul King, Jack Merner, Bobby Creech, Brad MacKenzie, Richard Waychoff i inni.
„w 1961 roku”	prawda z zastrzeżeniem	W 1961 roku powstał opis maszyny (biuletyn z lutego, artykuł z maja), ale pierwszy egzemplarz wysłano klientowi dopiero 1 kwietnia 1963 roku.
„maszyna stosowa”	prawda	Opisuje to Barton w 1961 roku, Levy w 1984 roku, a dla następcy — Hauck i Dent w 1968 roku.

Człon legendy	Werdykt	Dlaczego
„bez asemblera”	prawda częściowa	Użytkownik asemblera nie dostawał. Ale pierwszy, bębnowy MCP napisano w OSIL — „dość prymitywnym asemblerze” działającym na starszej maszynie Burroughs 220 — i zespół starał się to utrzymać w tajemnicy.
„MCP pierwszym systemem w języku wysokiego poziomu”	prawda z zastrzeżeniem	Dyskowy MCP dla B5500 pisano w ESPOL, odmianie ALGOL-u, wcześniej niż Multics w PL/I. Ale pierwszy MCP nie był w ESPOL.
„pierwsza pamięć wirtualna”	fałsz	Atlas w Manchesterze, ze stronicowaniem, uruchomiono uroczyście 7 grudnia 1962 roku, przed pierwszą dostawą B5000. Dla B5000 broni się najwyżej ostrożne „być może pierwszy komercyjny komputer z pamięcią wirtualną” — w wersji segmentowanej.
„znaczniki czyniły go odpornym na przepełnienia bufora”	prawda częściowa	Sprzęt sprawdzał granice tablic i odróżniał kod od danych. Ale B5000 miał jeden bit flagi, nie trzybitowe znaczniki, ochrona zależała od zaufanych kompilatorów, a w 1981 roku opisano udaną penetrację B6700.
„przetrwała jako Unisys ClearPath”	prawda	Zgodność kodu obiektowego od linii 5000, 6000 i 7000 przez A Series (1984) po ClearPath; wydanie MCP 20.0 z 2021 roku, mapa drogowa do 23.0.
„emulowana na Xeonach”	do doprecyzowania	Unisys mówi o warstwie oprogramowania układowego, która tłumaczy kod systemu na rozkazy x86, a w 2002 roku mówił o „wariancie maszyny wirtualnej”. Słowa „emulacja” w tych źródłach nie znalazłem.
„nazwę zapożyczył Tron”	nie da się ustalić	Nazwa „Master Control Program” stoi w dokumencie Burroughsa z lutego 1961 roku. Związek z filmem z 1982 roku jest kuszący, ale niepotwierdzony.

Oś tego tekstu: jedna nazwa, trzy rodziny, jedna linia, która przetrwała

Historia MCP to historia jednej nazwy noszonej przez kilka różnych systemów. Burroughs używał jej dla trzech wzajemnie niezgodnych rodzin maszyn: dużych (*Large Systems*), które zaczynają się od B5000, średnich (*Medium Systems*) z dziesiętną arytmetyką i małych (*Small Systems*) z mikroprogramowanym B1700. Do dziś dotrwała tylko pierwsza z nich. Jej droga prowadzi od B5000 (1961–1963) przez B5500 z dyskiem (1964), przez B6500 z trzybitowymi znacznikami i drzewem stosów (ogłoszony 1966, dostarczony 1969), B6700 i B7700 (1970), mikrokodowany B5900 (1980), A Series (1984), fuzję ze Sperry w Unisys (1986), ClearPath (1996), przejście na procesory Intela (zakończone w 2014 roku) aż do wersji wyłącznie programowej, która od 2016 roku działa w VMware, a później w chmurach Azure i AWS.

Po drodze trzy razy zmienił się język systemu: OSIL, ESPOL, NEWP.

Firma od maszyn sumujących (1885–1956)

St. Louis: arytmometr Williama Burroughsa

Zanim Burroughs zbudował komputer, przez siedemdziesiąt lat był firmą od liczenia pieniędzy. Według kompilacji z *International Directory of Company Histories*, udostępnionej w serwisie encyclopedia.com, William S. Burroughs w 1885 roku wynalazł pierwszą rejestrującą maszynę sumującą — urządzenie, które nie tylko dodawało, ale i drukowało wynik — nazwaną wówczas „arytmometrem” (*arithmometer*). W 1886 roku razem z trzema wspólnikami założył w St. Louis firmę **American Arithmometer Company**, która miała ten wynalazek sprzedawać.

Maszyna licząca Burroughsa w otwartym magazynie Computer History Museum — zdjęcie „Burrougł

Maszyna licząca Burroughsa w otwartym magazynie Computer History Museum — zdjęcie „Burroughs calculating machine” z 2010 roku. Źródło: Flickr, CC BY 2.0, autor: Marcin Wichary.

Z datą patentu jest pierwszy w tej historii kłopot. Źródła podają, że kluczowy patent Burroughsa pochodzi z 21 sierpnia 1888 roku albo że patent na model nadający się do sprzedaży uzyskano w 1892 roku[1]. Obie informacje mogą być prawdziwe jednocześnie, jeśli chodzi o dwa różne patenty — pierwszy na samą zasadę, drugi na wersję, którą dało się produkować seryjnie — ale żadne z moich źródeł tego nie rozstrzyga. W 1897 roku Burroughs dostał od Franklin Institute medal Johna Scotta, a rok później, w 1898 roku, zmarł na gruźlicę.

Detroit i Burroughs Adding Machine Company

Na początku XX wieku firma przeniosła się z St. Louis do Detroit i zmieniła nazwę na **Burroughs Adding Machine Company**. Źródła podają, że stało się to w 1904 albo w 1905 roku[2]; możliwe, że przeprowadzka nastąpiła w 1904 roku, a nowa nazwa — rok później, ale nie znalazłem dokumentu, który by to potwierdzał. W 1908 roku firma przejęła Universal Adding Machine i Pike Adding Machine, a w 1921 roku — Moon-Hopkins Billing Machine, producenta maszyn fakturujących.

Maszyna do dodawania Burroughs Klasse 3 („Pike”), Detroit, 1920 — model z linii przejętej od Pike A

Maszyna do dodawania Burroughs Klasse 3 („Pike”), Detroit, 1920 — model z linii przejętej od Pike Adding Machine Company. Źródło: Heinz Nixdorf MuseumsForum / museum-digital, CC BY-NC-SA, autor: Heinz Nixdorf MuseumsForum.

Ten rodowód ma znaczenie dla całej dalszej historii. Burroughs nie był firmą naukową ani wojskową, tylko dostawcą dla banków, księgowości i handlu. Jego klientami były instytucje, które liczą cudze pieniądze i nie mogą sobie pozwolić na pomyłkę. Kiedy pół wieku później maszyny z systemem MCP trafiały do banków w Kostaryce czy Nowej Zelandii, trafiały w środowisko, które firma znała od pokoleń.

Dziesięcioklawiszowa maszyna do dodawania z korbą Burroughs Portable, wyrób Burroughs Adding

Dziesięcioklawiszowa maszyna do dodawania z korbą Burroughs Portable, wyrób Burroughs Adding Machine Company z Detroit. Źródło: Industriemuseum Chemnitz / museum-digital, CC BY-NC-SA, autor: Industriemuseum Chemnitz, fot. Marion Kaiser.

Wejście w elektronikę: ElectroData i Pasadena

W 1953 roku firma przyjęła nazwę **Burroughs Corporation**. Według computinghistory.org.uk w czerwcu 1956 roku kupiła ElectroData Corporation z Pasadeny w Kalifornii, producenta maszyny Datatron 205, która pod marką Burroughsa stała się modelem **B205**. Kompilacja z encyclopedia.com łączy rok 1956 z pierwszym komercyjnym komputerem elektronicznym Burroughsa i z zakupem ElectroData. W 1959 roku upadł plan partnerstwa z RCA, po śmierci Johna Colemana.

Sprzedaż firmy rosła w tym czasie szybko: według tej samej kompilacji z 94 milionów dolarów w 1948 roku do 389 milionów w 1960 roku. Pasadena, gdzie po przejęciu ElectroData pracowali inżynierowie komputerowi, stała się miejscem, w którym narodził się B5000. Tam też, w ośrodku

Product Planning kierowanym przez W. R. Lonergana — od stycznia 1959 do kwietnia 1961 roku, jak wynika z listu dołączonego do wspomnień Waychoffa — zbierali się ludzie, o których jest następny rozdział.

W tej historii kilka razy wróci też inna maszyna firmy: **Burroughs 220**. Na niej działał asembler, w którym napisano pierwszy MCP, i na niej młody Alan Kay w lotnictwie USA po raz pierwszy zobaczył coś, co później nazwie zalążkiem idei obiektów. Paul Kimpel, autor emulatora B5500, napisał także emulatory ElectroData/Burroughs Datatron 205 i Burroughs 220 — oba opisuje jego strona phkimpel.us.

Pasadena 1958–1961: ludzie, zanim powstała maszyna

Skąd to wiemy: konferencja w Marina del Rey i opowieści Waychoffa

Prawie wszystko, co wiadomo o kulisach B5000, pochodzi z dwóch źródeł. Pierwsze to **Burroughs B 5000 Conference**, spotkanie zorganizowane 6 września 1985 roku w hotelu Marina del Rey w Kalifornii przez AFIPS i Burroughs Corporation, prowadzone przez Bernarda A. Gallera. Transkrypcja przechowywana jest w Charles Babbage Institute Uniwersytetu Minnesoty jako wywiad historii mówionej nr 98 (CBI OH 98). Przy stole siedzieli m.in. Robert (Bob) Barton, Lloyd Turner, David Dahm, Paul King, Richard Waychoff, Brad MacKenzie, Bobby Creech, Ben Dent, Clark Oliphint, John Hale i Jack Merner. Numery stron, które podaję, odnoszą się do tej transkrypcji.

Drugie źródło to tekst Richarda Waychoffa, współautora — razem z Lloydem Turnerem — kompilatora ALGOL dla B5000: *Stories About the B5000 and People Who Were There*, spisany około 1979 roku, z dopiskami z kwietnia 1985 roku, dostępny w zbiorach Eda Thelena. Jego liczby trzeba traktować jak wspomnienia, a nie dane firmowe.

Lato 1960: Knuth i kompilator za pięć tysięcy dolarów

Najsłynniejsza anegdota Waychoffa dotyczy studenta, który miał się stać jednym z najważniejszych informatyków stulecia. Latem 1960 roku Donald Knuth zaproponował Burroughsowi, że w czasie wakacji napisze kompilator ALGOL 58 dla B205 razem z podręcznikiem — za **5000 dolarów**. Gerard Guyod skomentował to podobno: *„We think that he already has it written”* („Myślimy, że on

już go ma napisany”).

Donald Knuth przed własnym cytatem na wystawie „Revolution” w Computer History Museum, 11 I 2

Donald Knuth przed własnym cytatem na wystawie „Revolution” w Computer History Museum, 11 I 2011 roku. Źródło: Flickr, CC BY 2.0, autor: Robert Scoble.

Kompilator Knutha czytał około 45 kart na minutę — dla porównania FORTRAN Waychoffa i Turnera czytał 40 — i zużywał przy tym mniej więcej połowę czasu maszyny. Nie zmieścił się na jednej szpuli taśmy papierowej; Brad MacKenzie zgodził się na dwie. Według Waychoffa Case Institute dał Knuthowi od razu magisterium zamiast licencjatu, i miał to być pierwszy taki przypadek w historii uczelni.

Bob Barton: sabotażysta przemysłowy

Postacią centralną legendy jest Robert S. Barton. David Dahm wspominał na konferencji w 1985 roku, że już latem 1958 roku Barton opisywał metodę tłumaczenia wyrażeń arytmetycznych na kod maszynowy, inspirowaną notacją logika, którego nazwisko transkrypcja zapisuje jako „Lukaszewicz” — chodzi o Jana Łukasiewicza. Działo się to, zanim obaj trafili do Burroughsa (s. 48). To pierwsze ogniwo łańcucha, który prowadzi od polskiej logiki do architektury stosowej; wracam do niego w rozdziale o artykule Bartona i w Wątku polskim.

Waychoff opisuje Bartona jako człowieka błyskotliwego i prowokującego. W styczniu 1960 roku w Dallas Barton miał oświadczyć, że IBM wydał 25 osobolat na FORTRANSIT, a „my trzej” zrobią to samo w pół roku. Na odczycie ACM w 1961 roku powiedział do około 1200 słuchaczy: *„there are only three people in this room that really know how to write a compiler”* — tylko trzy osoby na tej sali umieją pisać kompilatory — i wskazał Knutha, Turnera i Waychoffa.

W kwietniu 1960 roku, jak wspomina Waychoff, Barton wpadł o zmierzchu do biura Billa Lonergana i oznajmił, że pierwszy bęben magnetyczny w nowej maszynie *„cannot be optional”* — nie może być opcją. Lonergan zgodził się od razu, *„no arguments and no questions asked”*. Waychoff przypisuje Bartonowi pomysł pamięci wirtualnej — bęben był w nim niezbędny, bo to na nim miały czekać segmenty programu nieobecne w pamięci rdzeniowej.

Sam Barton na konferencji w 1985 roku nazwał się *„an industrial saboteur”* — sabotażystą przemysłowym (s. 50). Wpływał na projekt nieformalnie, z zewnątrz i z ukosa, a nie z pozycji szefa. Tak też podpisał swój artykuł z 1961 roku: jako *„computer consultant”* z Altadeny, z dopiskiem, że praca była sponsorowana przez Burroughs Corporation. W 1985 roku był ostry wobec maszyny, którą współtworzył: *„In my view, the machine got too big too soon”* — jego zdaniem maszyna urosła za bardzo i za szybko. Bębny w ogóle uważał za błąd: *„it made no sense to put those drums on the machine — no sense at all”*. Paul King oponował: bęben był wtedy *„the largest storage device available”*, największym dostępnym nośnikiem (s. 52–53).

Między relacją Waychoffa (Barton żąda bębna w 1960 roku) a słowami samego Bartona z 1985 roku (bębny nie miały sensu) jest sprzeczność, której nie umiem rozstrzygnąć; zostawiam obie relacje

obok siebie.

Zespół, który zbudował maszynę dla ALGOL-u

Uczestnicy konferencji z 1985 roku zgodnie mówią, że cel był jeden. Lloyd Turner: maszyna *„would execute ALGOL”*, a główne założenia ustalono do 1960 roku (s. 21–22). I dobitniej: *„We were building an ALGOL machine — pure and simple”* — budowaliśmy maszynę dla ALGOL-u, i tyle (s. 29). David Dahm: lista rozkazów *„was in fact designed to be the target of an ALGOL compiler”*, zaprojektowano ją jako cel kompilatora ALGOL-u (s. 32). Richard Waychoff tłumaczył wybór krótko: *„ALGOL was simply a better language”* (s. 31).

Burroughs ALGOL nie był czystym ALGOL-em 60. Był jego nadzbiorem, rozszerzonym m.in. o odwołania do pól słowa (*partial-word*), czyli do wybranych bitów w 48-bitowym słowie maszyny (s. 36–37).

Paul King wspominał, że wczesny projekt oparto na notacji polskiej, która *„implies a stack”* — z natury wymaga stosu. Na początku przewidziano nawet dwa stosy, osobny dla danych i osobny dla sterowania; połączył je w jeden dopiero Jack Merner (s. 46–50). Turner dodawał, że od początku miała to być *„multiprocessing, multiprogramming type of machine”* — maszyna wieloprocesorowa i wieloprogramowa (s. 22).

Ale ambicje wyprzedzały wiedzę. Dahm przyznał, że w 1961 roku przydział pamięci, pamięć wirtualna i wielozadaniowość były pojęciami mglistymi, które *„didn't become commercially viable”* — nie stały się opłacalne handlowo — mniej więcej do 1965 roku (s. 21). Brad MacKenzie wspominał, że specyfikacja kompilatorów i systemu operacyjnego miała się zmieścić w *„2000 words”* (s. 42). Ben Dent nie znalazł nigdzie listy, co MCP miał robić; zadanie systemu operacyjnego, jak to ujął, *„was very badly underestimated”* — zostało fatalnie niedoszacowane (s. 42).

Dlaczego w ogóle Burroughs zaryzykował coś tak nowego? Bobby Creech odpowiedział dwoma słowami, pytany, czy rozważano wtedy konkurencyjny system: *„No software”* — nie było oprogramowania (s. 13). Czytam to tak: firma nie miała bazy programów, którą musiałaby chronić, więc mogła zacząć od zera. Brak dorobku, który trzeba by przenosić na nową maszynę, był w tej sytuacji swobodą, nie tylko słabością.

Na konferencji padło też zdanie, które dobrze oddaje ideał projektantów. Bill Gant, wspominany przez uczestników, chciał, by system dało się uruchomić *„pushing no more than one button no more than once”* — naciskając nie więcej niż jeden przycisk nie więcej niż raz (s. 53).

„The Algol Roadmap” i zejścia rekurencyjne

Pracę nad kompilatorem poprzedziła praca nad samą składnią języka. Lloyd Turner narysował diagram składni ALGOL-u z dwoma kształtami pudełek — jednym dla definicji, drugim dla użycia pojęcia. Waychoff wspomina własną notację „pickles and boxes” — „ogórków i pudełek”. Według uczestników konferencji ta praca nad „mapą drogową” języka przyczyniła się do metody kompilacji przez zejścia rekurencyjne (*recursive descent*), w której każdej regule gramatyki odpowiada procedura kompilatora (s. 44–45).

Diagram zrobił karierę. Według Waychoffa dział marketingu Burroughsa wydrukował około 11 000 egzemplarzy, a czasopismo ACM rozesłało około 7000 — przy szacowanych wtedy około 14 000 programistach na całym świecie. Alan Perlis nazwał go „The Algol Roadmap” (1961). Niklaus Wirth użył później podobnej notacji diagramów w podręczniku Pascala.

Waychoff jest przy tym uczciwy wobec konkurencji: przyznaje, że A. A. Grau opisał zejścia rekurencyjne na sympozjum w listopadzie 1960 roku, około pięciu miesięcy przed „wynalazkiem” Waychoffa i Turnera.

Rok 1961: maszyna opisana, zanim zadziałała

Luty 1961: biuletyn „The Descriptor”

Najstarszym dokumentem B5000, jaki znalazłem, jest biuletyn Burroughsa nr 5000-20002-P z lutego 1961 roku, zatytułowany **„The Descriptor — A Definition of the B 5000 Information Processing System”**. Skan leży w zbiorach bitsavers. Ten dokument jest ważny z trzech powodów.

Po pierwsze, pada w nim już nazwa **„Master Control Program”** — tytuł rozdziału 4 (s. 4-1) — a skrót „MCP” raz, na s. 3-7. Nazwa systemu jest więc z 1961 roku, z dokumentu firmowego, sprzed jakiegokolwiek filmu. Po drugie, biuletyn opisuje już niemal wszystkie elementy, za które B5000 jest dziś ceniony: tablicę odniesień programu (PRT) o 1024 słowach, trzy typy deskryptorów (danych, wejścia-wyjścia i programu), bit obecności mówiący, czy dane są w pamięci rdzeniowej, czy na bębnie, w pełni relokowalne programy, 48-bitowe słowo i cztery 12-bitowe sylaby w słowie rozkazowym. Po trzecie — co dla legendy najważniejsze — biuletyn mówi wprost, że programuje się w ALGOL-u 60 albo w COBOL-u 61, a kodowania w języku maszynowym się nie używa (s. 3-5).

Dwa zdania z biuletynu dobrze oddają jego ton. Pierwsze: *„The B 5000 was designed as a complete system”* (s. 1-1) — B5000 zaprojektowano jako kompletny system, a nie jako procesor, do którego potem dopisuje się oprogramowanie. Drugie: *„The programmer need never know computer language to write an effective program for it”* (s. 2-8) — programista nie musi w ogóle znać języka maszyny, żeby napisać dla niej sprawny program. Biuletyn obiecuje też, że MCP zapewni jednoczesne operacje wejścia-wyjścia i „Multi-Processing” (s. 1-1), a drugi procesor da prawdziwą równoległość (s. 1-2). Na bębnie miały leżeć kompilator, system i segmenty programów.

Warto zauważyć ironię nazw. Biuletyn z lutego 1961 roku nosi tytuł „The Descriptor” — „Deskryptor” — a w artykule Bartona z maja tego samego roku słowo *descriptor* w ogóle nie pada. Barton mówi o słowach „lokalizatorach” (*locator*). Pojęcie, które dało tytuł dokumentowi firmowemu, w tekście głównego pomysłodawcy nosi inną nazwę.

Maj 1961: „A New Approach to the Functional Design of a Digital Computer”

Artykuł R. S. Bartona *„A New Approach to the Functional Design of a Digital Computer”* ukazał się w materiałach Western Joint Computer Conference (9–11 maja 1961 roku), w tomie 19 AFIPS Conference Proceedings, na s. 393–396. W 1987 roku przedrukowano go w „Annals of the History of Computing” (tom 9, nr 1, s. 11–15). Pod nazwiskiem autora stoi „computer consultant, Altadena, California”, a w tekście mowa o pracy sponsorowanej przez Burroughsa i o „nowym systemie przetwarzania informacji Burroughsa”. Mniej więcej w tym samym czasie, w maju 1961 roku, B5000 przedstawiło czasopismo „Datamation” (tom 7, nr 5, s. 28–32) — ten tekst przedrukowano później jako rozdział 22 w antologii Bella i Newella, jak podaje Douglas W. Jones.

Artykuł ma cztery strony i jest manifestem. Najważniejsze zdanie stoi na początku:

> *„Higher level programming languages, such as ALGOL, should be employed to the practical exclusion of machine language.”*

Języki wysokiego poziomu, takie jak ALGOL, powinny w praktyce całkowicie wyprzeć język maszynowy. Dalej Barton wyprowadza z tego konstrukcję maszyny. Skoro programy składają się z wyrażeń i procedur, to *„the concepts of expression and procedure lead directly to use of a Polish string program”* — pojęcia wyrażenia i procedury prowadzą wprost do programu w postaci „polskiego ciągu”. I zaraz wyjaśnia, skąd to określenie: *„The Polish notation was invented by the logician Lukasiewicz for use in the propositional calculus”* — notację polską wymyślił logik Łukasiewicz na potrzeby rachunku zdań. Jej zaletą, pisze Barton, jest to, że *„rules of operator precedence and signs of grouping are not required”* — nie potrzeba reguł pierwszeństwa działań ani nawiasów.

Z polskiego ciągu wynika stos: *„a vector of temporary storage cells referred to as the stack”*, wektor komórek pamięci tymczasowej nazywany stosem, który usuwa potrzebę jawnych rozkazów zapisu i pobrania wyników pośrednich. Każda deklaracja tablicy, przełącznika czy procedury dostaje *„a 'locator' word which is assigned in a table”* — słowo-lokalizator umieszczone w tablicy; to zalążek PRT. Słowa te zawierają informacje *„utilized in multi-programming and automatic segmentation control”* — wykorzystywane przy wieloprogramowości i automatycznym sterowaniu segmentacją. Dzięki temu, pisze Barton, *„we make a program invariant with respect to its position in storage”* — program staje się niezależny od miejsca w pamięci. I wreszcie zdanie, z którego wyrosną znaczniki: *„Each word in the stack has a control bit that distinguishes between values and names”* — każde słowo na stosie ma bit sterujący, który odróżnia wartości od nazw (czyli adresów).

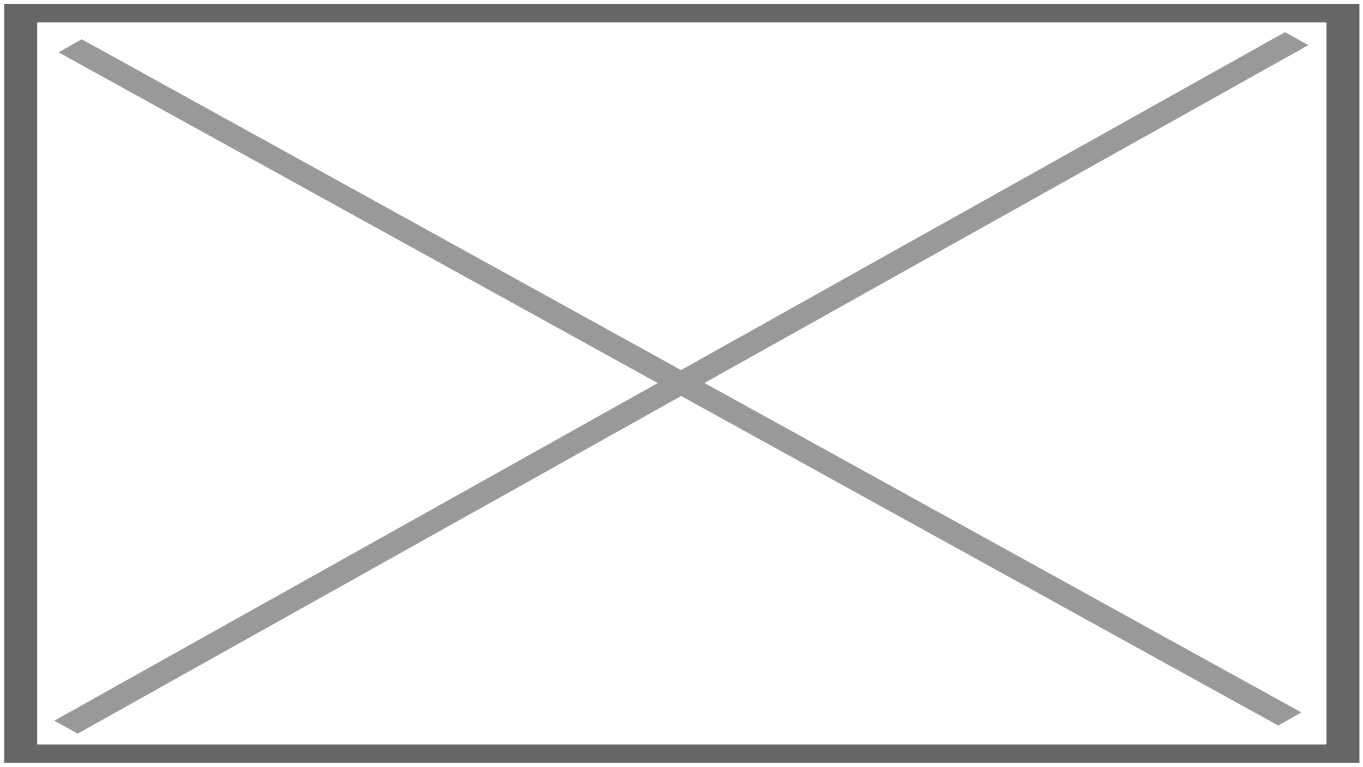
Z tego wszystkiego Barton wyciąga wniosek, który stanie się legendą: *„no lower level assembly-type language is required”* — nie jest potrzebny żaden język niższego poziomu typu asemblera. To zdanie jest programem, nie opisem stanu faktycznego. Za chwilę zobaczymy, że w 1961 roku i jeszcze przez kilka lat asembler był potrzebny — tylko nie użytkownikom.

Notacja polska i stos — jak to działa

Żeby zrozumieć, dlaczego notacja Łukasiewicza „implikuje stos”, jak to ujął Paul King, wystarczy jeden przykład. Przykład jest mój, nie Bartona, i pokazuje samą zasadę, a nie rzeczywisty kod B5000.

Weźmy wyrażenie `(a + b) × c`. W zwykłym zapisie potrzebujemy nawiasów, bo bez nich mnożenie wykonałoby się przed dodawaniem. Notacja Łukasiewicza opiera się na tym, że jeśli operator zapisze się zawsze po tej samej stronie swoich argumentów, nawiasy i reguły pierwszeństwa stają się zbędne. Maszyna stosowa korzysta z wariantu, w którym operator stoi *po* argumentach — to ten ciąg Barton nazywa polskim. Nasze wyrażenie przybiera postać `a b + c ×`.

Teraz maszyna czyta ten ciąg od lewej i stosuje dwie reguły: nazwę zmiennej zamienia na jej wartość i kładzie na wierzch stosu; operator zdejmuje z wierzchu dwie wartości, wykonuje działanie i kładzie wynik z powrotem. Kolejno: na stos trafia `a`, potem `b`; plus zdejmuje oba i kładzie `a+b`; na stos trafia `c`; razy zdejmuje `a+b` i `c` i kładzie wynik. Nigdzie nie trzeba było nazywać komórki na wynik pośredni ani pisać rozkazu „zapisz do pamięci” i „pobierz z pamięci”. Kompilator nie musi przydzielać miejsc na wyniki pośrednie — wystarcza wierzch stosu.



Animacja 1. *Notacja Łukasiewicza na stosie B5000. Kolejność sylab sama wyznacza kolejność działań, więc maszyna nie potrzebuje nawiasów, reguł pierwszeństwa ani komórek na wyniki pośrednie — wystarcza wierzch stosu.*

Kalkulatory HP z odwrotną notacją polską — operator wpisuje się po argumentach, jak w ciągu a b +

Kalkulatory HP z odwrotną notacją polską — operator wpisuje się po argumentach, jak w ciągu a b + c × — na Vintage Computer Festival Berlin 2014. Źródło: Flickr, CC0, autor: Wolfgang Stief.

W 1985 roku Barton powiedział, że B5500 jest interesujący, bo *„the code is so compact”* — kod jest tak zwarty (s. 52). Łańcuch prowadzi więc od logiki zdań przez sposób tłumaczenia wyrażeń, który Barton opisywał latem 1958 roku, po sprzęt.

Jeden szczegół warto zapamiętać na później. W polskim ciągu i na stosie leżą obok siebie dwa rodzaje rzeczy: *wartości* (liczby) i *nazwy* (adresy zmiennych, tablic, procedur). Maszyna musi wiedzieć, z którym ma do czynienia, bo z liczbą robi się co innego niż z adresem. Bit sterujący, o którym pisze Barton, rozróżnia właśnie te dwa przypadki. Z tego jednego bitu wyrośnie w następnych latach cała historia znaczników w pamięci Burroughsa.

Jak działał B5000 od środka

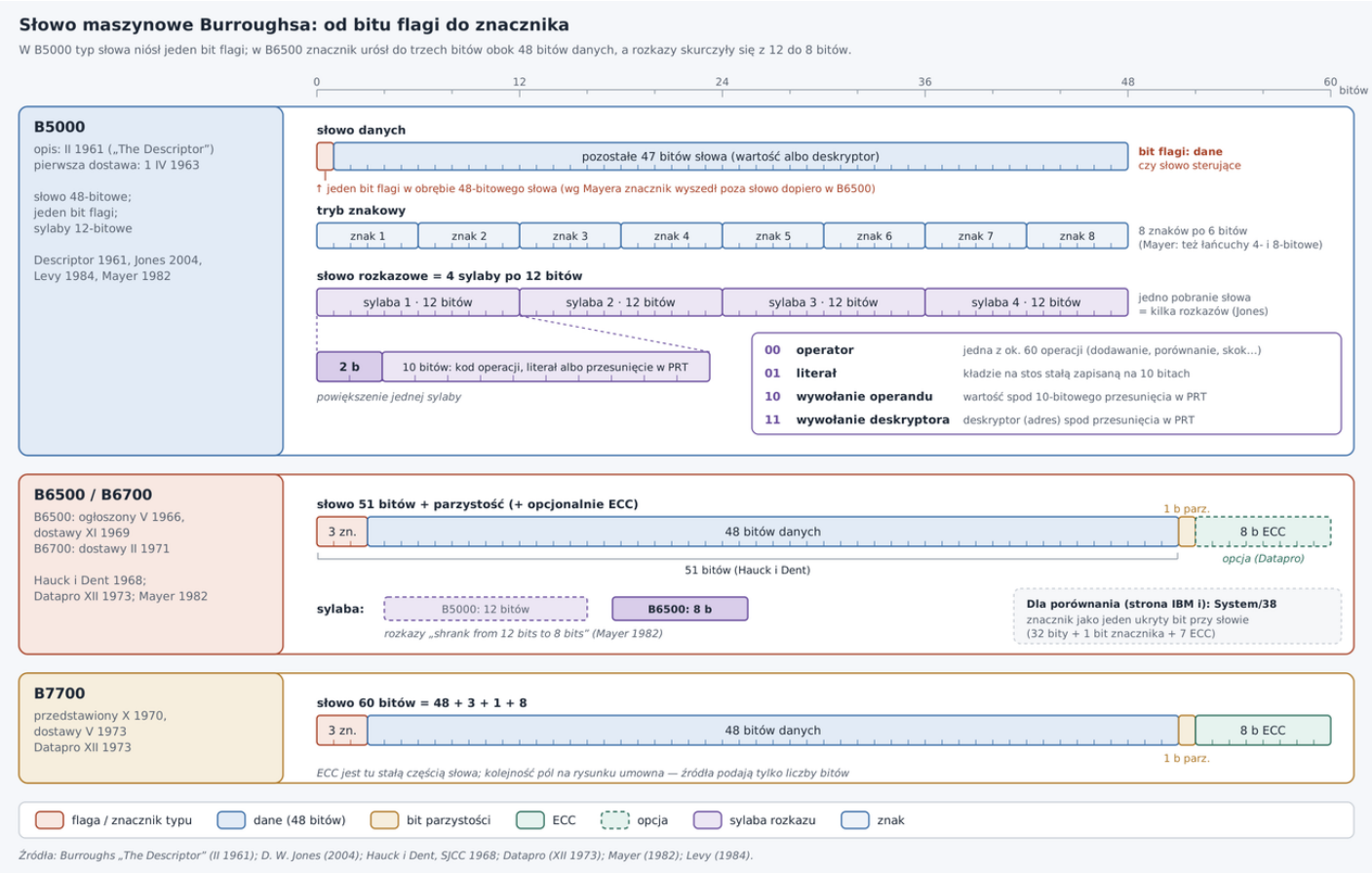
Słowo, sylaby i tryb znakowy

Podstawową jednostką pamięci B5000 jest **słowo 48-bitowe**. Słowo z rozkazami dzieli się na **cztery sylaby po 12 bitów** — to dane z biuletynu z 1961 roku i z rozdziału o B5000 w książce Henry'ego M. Levy'ego *Capability-Based Computer Systems* (1984). Douglas W. Jones z University of Iowa w notatkach do wykładu z 2004 roku opisuje, jak sylaba mówi, czym jest: o tym decydują jej dwa pierwsze bity.

Dwa pierwsze bity sylaby	Rodzaj	Co robi (wg D. W. Jonesa)
00	operator	jedna z około 60 operacji, np. dodawanie, porównanie, skok
01	literał	kładzie na stos stałą zapisaną na 10 bitach
10	wywołanie operandu	pobiera wartość spod 10-bitowego przesunięcia w PRT
11	wywołanie deskryptora	kładzie na stos deskryptor (adres) spod przesunięcia w PRT

Ta tabela to polski ciąg zamieniony na bity. Literał i wywołanie operandu odpowiadają kładzeniu wartości na stos, wywołanie deskryptora — kładzeniu nazwy, a operator — działaniu na wierzchu stosu. Jones zauważa, że jedno pobranie słowa z pamięci dostarcza do trzech rozkazów; o zwartości kodu Barton mówił z dumą jeszcze w 1985 roku.

Obok trybu słowowego B5000 miał **tryb znakowy**: według Jonesa w słowie mieściło się wtedy osiem znaków 6-bitowych. Mayer w 1982 roku dodaje, że maszyna obsługiwała tablice słów albo łańcuchy znaków 4-, 6- lub 8-bitowych. Jones zwraca też uwagę na rzecz rzadko spotykaną: liczby całkowite i zmiennoprzecinkowe miały w B5000 jeden wspólny format.



Rysunek 1. *Od jednego bitu flagi w B5000 do trzech bitów znacznika w B6500 i 60-bitowego słowa B7700: znacznik wyszedł poza 48 bitów danych, a rozkazy skurczyły się z 12 do 8 bitów.*

PRT: tablica, która jest całym światem programu

Każdy program na B5000 miał własną **tablicę odniesień programu** (*Program Reference Table*, PRT), o długości do 1024 słów 48-bitowych. Jej adres trzymał rejestr sprzętowy. Levy pisze, że *„the Program Reference Table completely defines the domain of execution for each user program"* — PRT w pełni określa dziedzinę wykonania każdego programu. Program mógł sięgnąć tylko po to, co jest w jego PRT: zmienne proste (skalary) i deskryptory tablic, procedur i urządzeń.

Stąd 10-bitowe przesunięcia w sylabach: dziesięć bitów to dokładnie 1024 możliwe pozycje. Rozkaz nie zawiera adresu w pamięci, tylko numer pozycji w PRT. To właśnie miał na myśli Barton, pisząc o programie niezależnym od położenia w pamięci: kod mówi „weź to, co leży w pozycji 17", a gdzie fizycznie są dane, wie tylko słowo w pozycji 17.

Każde słowo w PRT niosło **znacznik** — w B5000 jeden bit — który według Levy'ego *„indicates whether the entry is a descriptor or a scalar data element"*: mówi, czy wpis jest deskryptorem, czy skalarem z danymi.

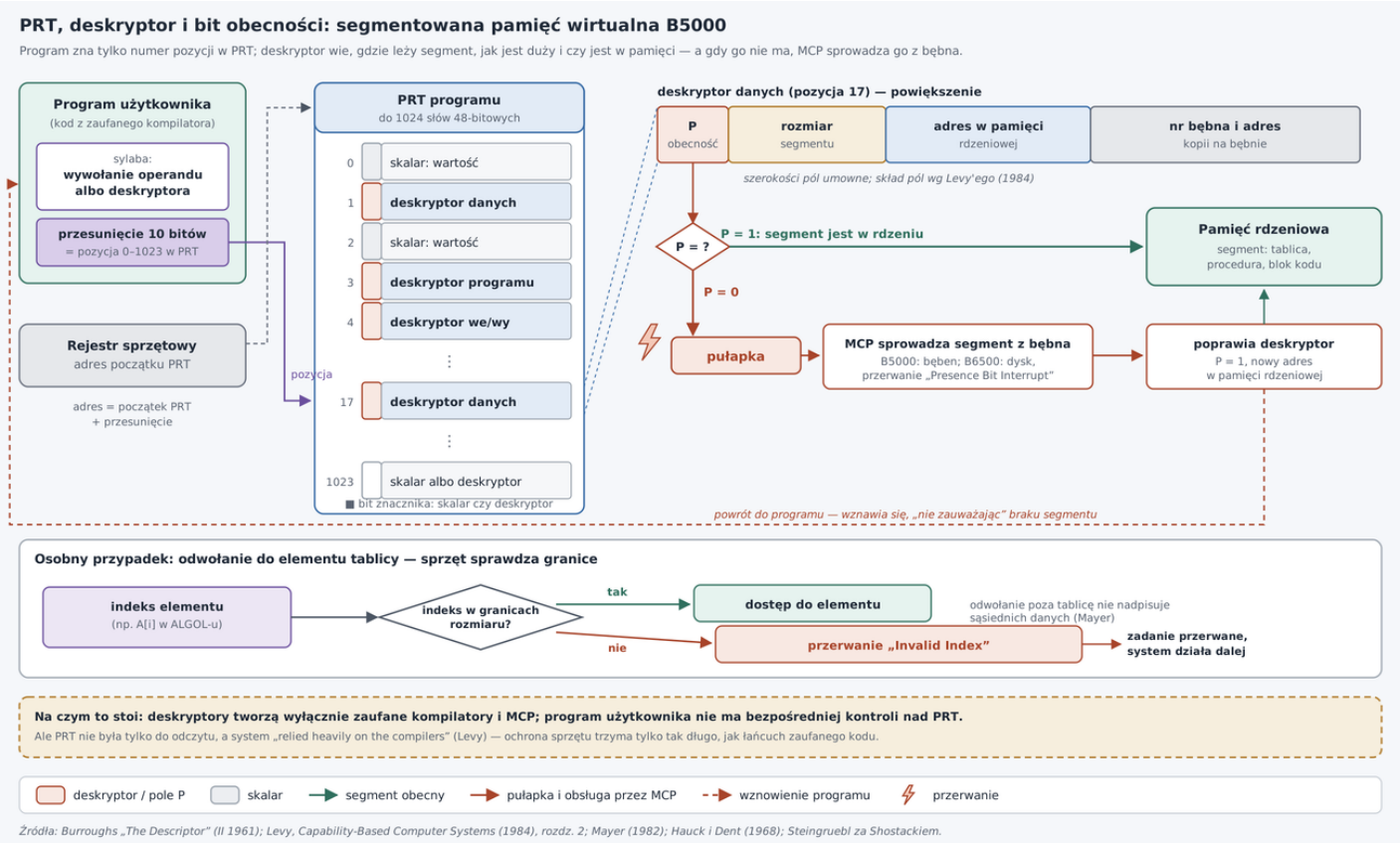
Deskryptory i bit obecności

Deskryptor to słowo, które opisuje obszar pamięci zamiast go zawierać. Levy wyróżnia trzy rodzaje, te same co biuletyn z 1961 roku: deskryptory danych, programu i wejścia-wyjścia. Deskryptor danych zawiera *„the size, primary memory address, and drum unit number and address of a data segment”* — rozmiar segmentu, jego adres w pamięci głównej oraz numer i adres bębna, na którym leży kopia.

Najważniejszy jest jeden bit: **bit obecności** (*presence bit*, P). Jeśli jest ustawiony, segment jest w pamięci rdzeniowej i adres w deskryptorze jest ważny. Jeśli nie — próba sięgnięcia do segmentu powoduje pułapkę, MCP przejmuje sterowanie, sprowadza segment z bębna, poprawia deskryptor i wznawia program, który nawet tego nie zauważa. To jest segmentowana pamięć wirtualna na żądanie — w wersji z 1961 roku. Mayer w 1982 roku nazywa bit obecności podstawą pamięci wirtualnej B5000.

Deskryptor niesie też rozmiar segmentu. Dzięki temu każde odwołanie do elementu tablicy sprzęt może sprawdzić z rozmiarem — Mayer pisze, że deskryptor *„allows bounds checking (done automatically by the hardware)”*, pozwala na sprawdzanie granic wykonywane automatycznie przez sprzęt. Odwołanie poza tablicę nie nadpisuje sąsiednich danych, tylko kończy się przerwaniem.

Deskryptor programu działa jeszcze inaczej: według Levy'ego samo odwołanie do niego automatycznie wywołuje procedurę. O tym, co się stanie przy odwołaniu, decyduje więc rodzaj słowa w PRT, a nie rozkaz w programie.



Rysunek 2. *Program nie zna adresów: zna tylko numer pozycji w PRT. To deskryptor wie, gdzie leży segment, jak jest duży i czy w ogóle jest w pamięci — a gdy go nie ma, MCP sprowadza go z bębna, zanim program to zauważy.*

Stos w rejestrach A i B — i jego słaby punkt

Stos B5000 nie leżał w całości w procesorze. Levy opisuje, że dwa wierzchołkowe elementy trzymano w rejestrach A i B, a reszta stosu była w pamięci.

Ten projekt miał słabość, którą Levy wskazuje wprost. Deskryptory mogły być kopiowane na stos. Wtedy deskryptor istnieje w dwóch miejscach: w PRT i na stosie. Gdy MCP chciał przenieść segment w inne miejsce pamięci, musiał poprawić wszystkie deskryptory, które na niego wskazują — a więc przeszukać stos. W B6500 deskryptor dostał osobny bit oznaczający, że jest kopią deskryptora-matki — o tym niżej.

Wieloprocesorowość i we/wy

Jones wylicza w swoich notatkach innowacje, które B5000 zebrał w jednej maszynie: relokację w czasie wykonania, segmentowaną pamięć wirtualną, czystą architekturę stosową, wejście-wyjście sterowane przerwaniami, rejestry wierzchołka stosu oraz wieloprocesorowość przez przełącznicę krzyżową i pamięć wieloportową. Jones dodaje, że w latach siedemdziesiątych Burroughs sprzedawał maszyny z sześcioma procesorami, podczas gdy IBM ostrożnie trzymał się dwóch.

Taka maszyna nie mogła działać bez systemu operacyjnego. Program użytkownika nie miał ani rozkazów do zarządzania pamięcią, ani do sterowania urządzeniami, ani dostępu do cudzych PRT. Wszystko to musiał robić MCP — i to on, a nie sprzęt, okazał się najtrudniejszą częścią projektu.

Pamięć wirtualna: B5000 czy Atlas

Spór o słowo „pierwszy”

Zdanie „B5000 był pierwszym komputerem z pamięcią wirtualną” powtarza się często, ale nie wytrzymuje zestawienia dat. W Manchesterze zespół Toma Kilburna zbudował razem z firmą Ferranti komputer **Atlas**, który uroczyście uruchomiono **7 grudnia 1962 roku**. Tablica IEEE Milestone, odsłonięta 21 czerwca 2022 roku, mówi bez ogródek: *„The most significant new feature of Atlas was the invention of virtual memory”* — najważniejszą nowością Atlasa był wynalazek pamięci wirtualnej — i dodaje, że *„Virtual memory became a standard feature of general-purpose*

computers”. Atlas stosował stronicowanie, czyli podział pamięci na strony o stałej długości.

B5000 ma za sobą opis pamięci segmentowanej z bitem obecności już w biuletynie z lutego 1961 roku, ale pierwszy egzemplarz trafił do klienta dopiero 1 kwietnia 1963 roku — cztery miesiące po inauguracji Atlasa. Dla „pierwszeństwa” w sensie działającej maszyny Atlas wygrywa. W sensie idei sprawa jest mniej jasna, bo oba projekty powstawały mniej więcej w tym samym czasie, a żadne z moich źródeł nie twierdzi, że jeden zespół wiedział o drugim.

Źródła formułują pierwszeństwo B5000 ostrożnie albo w ogóle go nie przyznają[3]. Mayer w 1982 roku pisze, że B5000 był *„perhaps the first commercial computer to provide virtual memory”* — „być może pierwszym komercyjnym komputerem z pamięcią wirtualną”. Polskie kalendarium Zbigniewa Ryznara mówi o „pierwszym komercyjnym komputerze z segmentowaną pamięcią wirtualną”. IEEE przypisuje sam wynalazek Atlasowi. Do tego dochodzi zastrzeżenie, które osłabia nawet ostrożną wersję: Atlas też był produktem handlowym Ferranti, więc „pierwszy komercyjny z pamięcią wirtualną w ogóle” jest dla B5000 wątpliwy. Najbezpieczniej brzmi więc teza najwęższa: **B5000 był pierwszą komercyjną maszyną z pamięcią wirtualną opartą na segmentach**, a nie na stronach.

Segmenty a strony

Różnica nie jest tylko terminologiczna. W Atlasie strona to kawałek pamięci o stałej długości, niezwiązany z tym, co program w nim trzyma. W B5000 segment to jednostka logiczna: jedna tablica, jedna procedura, jeden blok kodu — kompilator dzieli program na segmenty tak, jak program dzieli się na części. Deskryptor zna rozmiar segmentu, więc sprawdzanie granic przychodzi przy okazji. Datapro w raporcie o B6700 z 1973 roku opisuje tę samą zasadę jako pamięć wirtualną przez „segmentację logiczną”: kompilatory dzielą programy na segmenty, a MCP je ładuje i przenosi.

Tę linię — segmenty jako jednostki logiczne, widoczne dla programu — podjął później Multics. Jego historię i własny spór o „pierwszeństwa” opisuje strona **Multics**.

Entuzjazm studzi sam zespół: według Dahma pamięć wirtualna stała się opłacalna handlowo dopiero około 1965 roku, czyli w czasach B5500 z dyskiem i nowym MCP. Pierwszy B5000 miał ją w architekturze, ale jej działanie zależało od systemu, który w 1963 roku był daleki od gotowości.

MCP: system, którego nikt nie wycenił

Nazwa z 1961 roku

Nazwa **Master Control Program** — „główny program sterujący” — pochodzi z dokumentu Burroughsa z lutego 1961 roku. Opisuje to, czym system miał być: programem, który steruje wszystkim, czego programy użytkowe robić nie mogą. W biuletynie „The Descriptor” MCP odpowiada za jednoczesne operacje wejścia-wyjścia i wieloprzetwarzanie, a z opisu deskryptorów wynika, że to on sprowadza nieobecne segmenty z bębna.

Burroughs używał tej nazwy nie tylko dla linii B5000; Unisys nosi ją do dziś w nazwie ClearPath MCP.

Bębnowy MCP i OSIL: tajemnica zespołu

Tu legenda o maszynie „bez asemblera” trafia na najtwardszy dowód. Na konferencji w 1985 roku Brad MacKenzie powiedział, że pierwszy MCP — tak zwany **bębnowy MCP** (*drum MCP*), działający na B5000 z bębnem magnetycznym — napisano w języku **OSIL**, który określił jako *„a pretty primitive assembler”*, dość prymitywny asembler. I dodał zdanie, które wiele mówi o tym, jak ważna była dla zespołu zasada Bartona: *„We tried hard to keep it a secret”* — bardzo staraliśmy się utrzymać to w tajemnicy (s. 38).

Bobby Creech uzupełnił: ten asembler nie działał na B5000, tylko na **Burroughs 220** (s. 38). Pierwszy system operacyjny maszyny, która miała nie potrzebować asemblera, powstał więc przy użyciu asemblera innej, starszej maszyny. Rozwinięcia nazwy OSIL w transkrypcji nie znalazłem.

Barton w 1985 roku nie wypierał się tego, tylko tłumaczył. Zasada wymagała, żeby *„the operating system implemented in a higher-level language”* — system operacyjny był zrealizowany w języku wyższego poziomu — ale o dyskowym MCP, który tę zasadę spełnił, powiedział: *„somebody had to do something first and they had to learn a lot”* — ktoś musiał najpierw coś zrobić i dużo się przy tym nauczyć. Dodał też, że próbowano tego *„a year before, but the notorious Erdwin refused”* — rok wcześniej, ale „osławiony Erdwin” odmówił (s. 54).

1 kwietnia 1963: „Nothing worked”

Datę pierwszej dostawy podał na konferencji Brad MacKenzie: *„the first shipment was on April Fool's Day, 1963”* — w prima aprilis, 1 kwietnia 1963 roku. Pierwszym klientem było UTC, drugim NASA; NASA była zarazem pierwszym klientem, który maszynę dzierżawił (s. 26, 41). Rozwinięcia skrótu UTC transkrypcja nie podaje; „United Technology Center” nasuwa się samo, ale tego nie potwierdziłem. Między opisem maszyny a pierwszą dostawą minęły ponad dwa lata, więc gdy czyta się, że B5000 „powstał w 1961 roku”, trzeba pamiętać, że źródła łączą z nim lata od 1961 do 1963 [4].

O tym, jak wyglądał wtedy system, uczestnicy mówili bez złudzeń. Clark Oliphint wspominał, że MCP *„fell behind”* — nie nadążał — a przydział taśm i wieloprogramowość były na początku

niekompletne (s. 40–41). John Hale ujął to w dwóch słowach: *„Nothing worked”* — nic nie działało (s. 40). Pierwsze wydanie kompilatora COBOL dla B5000 pojawiło się około października 1963 roku (s. 41).

Waychoff opowiada, jak sprzedawano B5000 jeszcze przed dostawą. W 1962 roku Gene Thompson z UTC postawił warunek: bez FORTRAN-u nie będzie B5000 w UTC. Waychoff rozerwał wtedy przy nim nieotwartą kopertę z podręcznikiem ALGOL-u i powiedział: *„Maybe you should read your mail sometime”* — „może czasem warto przeczytać pocztę”. Sprzedaż doszła do skutku. Ulubionym klientem B5000 był według Waychoffa Uniwersytet Stanforda, który dostał IBM 7090 i dar 900 000 dolarów, z czego 500 000 wydał na B5000.

Pięć Corvette za nowy MCP

Waychoff opisuje też niezwykłą propozycję zespołu kompilatora. Lloyd Turner, który według Waychoffa zarabiał 900 dolarów miesięcznie, zaproponował, że jego ludzie przepiszą MCP po nocach i w weekendy — jeśli Burroughs kupi im pięć Corvette po 5000 dolarów. Kierownictwo odmówiło.

Dysk z głowicą na ścieżkę i dyskowy MCP

Pretekst pojawił się wraz z nowym nośnikiem. Według Waychoffa dysk z głowicą na każdą ścieżkę dał Bradowi MacKenzie powód, by zespół Turnera napisał nowy MCP — tym razem dla dysku, nie dla bębna. Waychoff nie ma wątpliwości, że bez tego B5000 *„would have clearly failed”* — oczywiście by przepadł. Gdy system dyskowy ukończono, maszynę ogłoszono ponownie jako **B5500**.

Ten nowy system, nazywany w dokumentach „DF MCP” (*Disk File MCP*), pisano już w języku wysokiego poziomu — w **ESPOL**. Paul King wspominał, że w ESPOL napisano m.in. obsługę plików dyskowych, tzw. *disk filer* (s. 39).

ESPOL: ALGOL dla systemu, w trzy tygodnie

Skąd wziął się ESPOL? Lloyd Turner opowiadał, że powstał z kompilatora „extended ALGOL” w mniej więcej **trzy tygodnie**. Dodano do niego dostęp do pamięci „jak do tablicy”, niezbędny do jej przydzielania (s. 38).

Nazwę rozwija podręcznik *B5500 ESPOL Reference Manual*, oznaczony © 1966 Burroughs Corporation, w wydaniu z października 1967 roku: **Executive Systems Problem Oriented Language**. Podręcznik mówi, że ESPOL *„was designed to include many features of Extended ALGOL”* i że programy w Extended ALGOL nie służą do obsługi przerwań ani przydziału pamięci —

to zadanie DF MCP. Cały opis ESPOL jest ułożony w kategoriach konstrukcji Extended ALGOL, a we wstępie stoi ostrzeżenie: *„Use of this document requires a thorough knowledge of Extended ALGOL”* — korzystanie z dokumentu wymaga gruntownej znajomości Extended ALGOL.

Mayer w 1982 roku pisał, że MCP był pisany w *„a HLL, ESPOL (an ALGOL variant)”* — w języku wysokiego poziomu ESPOL, odmianie ALGOL-u. To zdanie jest prawdziwe dla MCP od B5500. Nie jest prawdziwe dla pierwszego MCP z 1963 roku.

> **Werdykt w sprawie „bez asemblera”.** Użytkownik B5000 nie dostawał asemblera i według biuletynu z 1961 roku nie potrzebował go: programował w ALGOL-u albo COBOL-u. Pierwszy MCP powstał jednak w OSIL, asemblerze działającym na Burroughs 220, a zespół trzymał to w tajemnicy. Dopiero dyskowy MCP dla B5500 napisano w ESPOL. Zdanie „B5000 nie miał asemblera” jest więc prawdziwe dla klienta, a fałszywe dla twórców systemu.

Kompilator ALGOL-u: najszybszy na świecie i jego zmora

Waychoff nie ukrywa dumy z kompilatora. W 1963 roku był to według niego *„the best compiler on earth”* — najlepszy kompilator na Ziemi — przetwarzający około 1750 linii na minutę. Zespół miał też poczucie humoru: procedurę o nazwie MONITOR nazwano w kompilatorze MERRIMAC, z komentarzem *„THIS TIME THE MERRIMAC WILL HANDLE THE MONITOR”*. To gra słów nawiązująca do bitwy pancerników Monitor i Merrimack.

Była też zmora. Waychoff nazywa „procedury strumieniowe” (*Stream Procedures*) *„the biggest blight on the B5000 Algol compiler”* — największą plagą kompilatora. Był to sposób połączenia trybu znakowego maszyny z ALGOL-em: fragmenty programu działające na znakach, a nie na słowach. Dlaczego tak ich nie lubił, w moich wypisach z jego tekstu nie ma.

Wizyta Dijkstry

Jedna z najlepszych anegdot Waychoffa dotyczy Edsgera Dijkstry. Podczas jego wizyty program złożony z samego `BEGIN END.` skompilował się i wykonał w niecałe trzy sekundy. Dijkstra miał powiedzieć: *„My God. How does one get any conception of how fast it is?”* — „Mój Boże. Jak w ogóle można pojąć, jak to jest szybkie?”. Przy tej samej okazji Dijkstra w około cztery minuty rozprawił się z deklaracją DEFINE, o którą zespół spierał się wewnętrznie przez cztery dni.

Waychoff opowiada też, że przyjaciele Dijkstry w Europie chcieli kupić trzy B5000, ale wiceprezes grupy międzynarodowej Burroughsa, Ray Macdonald, zablokował sprzedaż z powodu kosztów serwisu trzech maszyn.

B5500 i B5700: maszyna, która zaczęła się sprzedawać

Ile sprzedano B5000

Zanim B5500 uratował linię, B5000 był handlową porażką — przynajmniej według Waychoffa, który jako jedyny z moich źródeł podaje liczby. Próg rentowności miał wynosić **35 maszyn**. W pierwszym roku sprzedano 17, w drugim 15, w trzecim — jedną. To są liczby ze wspomnień, nie z dokumentów firmy, i trzeba je czytać jako rząd wielkości. Pokazują jednak coś, czego legenda nie mówi: maszyna, która „wyprzedzała swój czas", przez pierwsze lata nie zarabiała.

Barton w 1985 roku podsumował to gorzko: *„there really was no recognition of what was good about that machine for at least eight years"* — przez co najmniej osiem lat nikt nie dostrzegał, co w tej maszynie było dobre (s. 54). Za niedocenionych bohaterów uznał *„the group of amateurs who worked on the ALGOL implementation language"* — grupę amatorów, którzy pracowali nad językiem implementacji opartym na ALGOL-u (s. 55).

11 sierpnia 1964: B5500

Komunikat prasowy Burroughsa z **11 sierpnia 1964 roku** ogłaszał B5500 jako maszynę o *„up to three times more productive capacity than the B5000"* — do trzech razy większej wydajności produkcyjnej niż B5000. Sercem nowości był dysk z głowicą na ścieżkę: dostęp do dowolnego rekordu trwał średnio 1/50 sekundy, jedna jednostka mieściła 9,6 miliona znaków, a cały podsystem — prawie miliard znaków.

Ceny z tego samego komunikatu: pełny B5500 można było dzierżawić za mniej niż 20 000 dolarów miesięcznie albo kupić za 830 000 dolarów; typowa konfiguracja dwuprocesorowa kosztowała około 35 000 dolarów miesięcznie albo 1 473 000 dolarów; maksymalna — ponad 5 milionów. Pokazy w Pasadenie zapowiedziano od października 1964 roku, a dostawy — pół roku po ogłoszeniu.

Mayer w 1982 roku datuje B5500 na 1964 rok. W dokumentach z epoki, które przechowuje bitsavers, widać, jak system dojrzewał: *Narrative Description of B5500 MCP* (październik 1966), *B5500 DISK MCP Handbook* (październik 1967), wspomniany *B5500 ESPOL Reference Manual* (październik 1967) i podręcznik *Operating the B5500 Time-Sharing System*.

B5700, wydania Mark i podział czasu

Ostatnim wcieleniem linii 5000 był **B5700**. Z dokumentów w bitsavers wynika, że miał własny system podziału czasu (*B5700 TSS Reference Manual*, wrzesień 1972), a jego oprogramowanie

systemowe ukazywało się w wydaniach numerowanych „Mark”. Wydanie **Mark XIII** z 1971 roku jest dziś dostępne w emulatorze retro-b5500, a dokumenty *B5700 Mark XV.3.0 System Note #13* (lipiec 1974) i *B5700 Mark XVI.0 System Note #14* (czerwiec 1975) pokazują, że po Mark XIII były jeszcze co najmniej Mark XV i XVI.

W 1973 roku Elliott I. Organick wydał książkę *Computer System Organization: The B5700/B6700 Series*, której skan w całości leży w bitsavers.

B6500, B6700, B7700: znaczniki i drzewo stosów

Ogłoszenie, opóźnienie i maszyna, której nie dostarczono

Następca linii 5000 nie był prostą rozbudową B5500. Według raportu Datapro z grudnia 1973 roku **B6500** ogłoszono w maju 1966 roku, specyfikację zmieniono w lipcu 1967 roku, a pierwsze dostawy opóźniły się do **listopada 1969 roku** — trzy i pół roku od ogłoszenia. Większy **B7500** (w wersjach B7504 i B7506) ogłoszono w lipcu 1967 roku, ale **nigdy go nie dostarczono**; zastąpił go B7700.

W październiku 1970 roku Burroughs przedstawił **B6700** i **B7700**. B6700 trafiał do klientów od lutego 1971 roku, B7700 — od maja 1973 roku. W czerwcu 1973 roku ogłoszono B6748, z dostawami od pierwszego kwartału 1974 roku. Do grudnia 1973 roku na świecie pracowało według Datapro około **150 B6700**, z czego około 60 w USA, i zaledwie **dwa B7700**.

Waychoff łączy drogę do B6500 z Bartonem: to jego *rejestry display*, służące do adresowania zmiennych z wyższych poziomów zagnieżdżenia procedur, miały wyznaczyć kierunek. Barton w 1961 roku wyobrażał sobie też maszynę złożoną z wyspecjalizowanych procesorów porozumiewających się przez mechanizm sterowania, który nazwał „Sesame” — od „Sezamie, otwórz się”.

Przy projektowaniu B6500 zdarzyła się anegdota, którą opowiada Waychoff. Konsultant Bill McKeeman deklarował, że jego metoda kompilacji osiągnie 40 000 linii na minutę. Bob Novak zauważył, że maszyna czyta około 4000 linii na minutę — dziesięć razy mniej. Zanim to zauważono, Burroughs zdążył uruchomić projekt na podstawie tamtej obietnicy.

Trzy bity znacznika

Najważniejszą zmianą w B6500 było słowo. W artykule *„Burroughs' B6500/B7500 Stack Mechanism”* z konferencji SJCC 1968 (s. 245–251) E. A. Hauck i B. A. Dent opisują **słowo 51-bitowe**: **3 bity znacznika i 48 bitów danych**. Mayer w 1982 roku ujmuje tę samą zmianę jako rozszerzenie bitu flagi B5000 do trzybitowego znacznika leżącego *poza* 48-bitowym słowem, a przy okazji odnotowuje, że rozkazy *„shrank from 12 bits to 8 bits”* — skurczyły się z 12 do 8 bitów.

Znaczniki identyfikują typ słowa. Dzięki temu, piszą Hauck i Dent, sprzęt automatycznie obsługuje operandy jedno- i dwusłowowe (pojedynczej i podwójnej precyzji) i chroni kod na poziomie pojedynczego słowa. Próba wykonania danych jako kodu albo zmodyfikowania kodu kończy się przerwaniem.

Datapro w 1973 roku opisuje słowo B6700 nieco pełniej: 48 bitów danych, 3 bity kontrolne, 1 bit parzystości i opcjonalnie 8 bitów kodu korekcji błędów (ECC). B7700 miał słowo **60-bitowe**: 48 + 3 + 1 + 8.

Deskryptory danych i segmentów

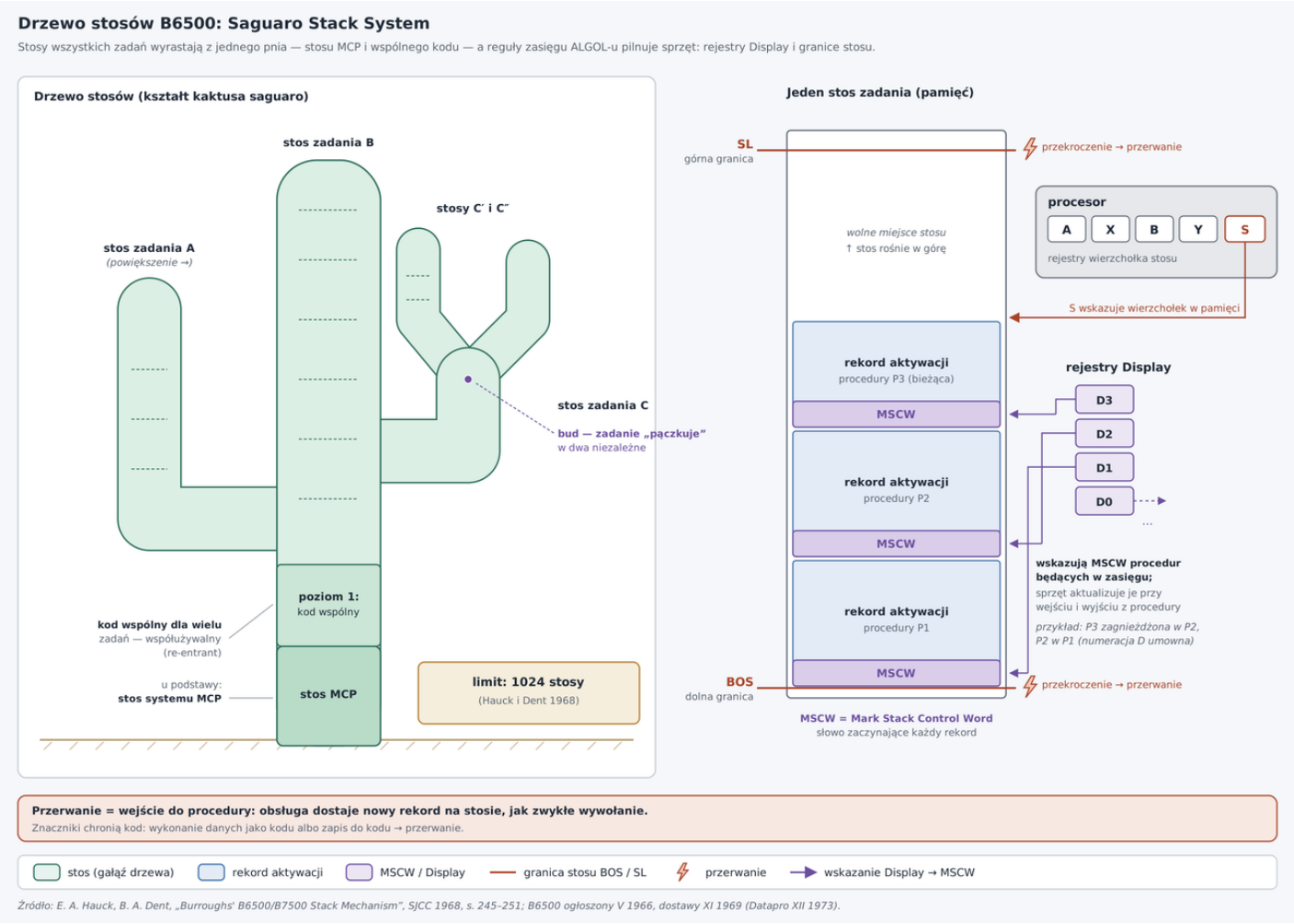
Deskryptory przeszły do B6500 z B5500. Hauck i Dent wymieniają deskryptory danych i segmentów (DD i SD) z 20-bitowym polem adresu bezwzględnego. Bit obecności P działa jak wcześniej, ale jego przerwanie ma teraz nazwę — *Presence Bit Interrupt* — i MCP sprowadza brakujące dane już nie z bębna, tylko z dysku. Do deskryptora doszły kolejne bity: I (deskryptor indeksowany), D (podwójna precyzja), R (tylko do odczytu) i C (kopia deskryptora-matki). Ochrona przed zapisem jest więc cechą deskryptora, a kopię da się odróżnić od oryginału.

Saguaro Stack System

Najbardziej oryginalną częścią B6500 jest organizacja stosów. Hauck i Dent opisują cztery rejestry wierzchołka stosu — A, X, B i Y — oraz rejestr S. Każde zadanie ma własny stos z dolną granicą BOS i górną SL; przekroczenie granicy wywołuje przerwanie. Stos składa się z rekordów aktywacji procedur, a każdy zaczyna się słowem sterującym MSCW (*Mark Stack Control Word*). Rejestry Display wskazują MSCW wszystkich procedur, które są w danej chwili w zasięgu, a sprzęt sam aktualizuje je przy wejściu do procedury i przy wyjściu z niej. To jest sprzętowa realizacja reguł zasięgu ALGOL-u: procedura zagnieżdżona widzi zmienne procedur, w których ją zadeklarowano.

Stosy wszystkich zadań tworzą drzewo. Hauck i Dent nazwali je **„Saguaro Stack System”** — od kaktusa saguaro, którego ramiona wyrastają z pnia; dziś mówi się zwykle o stosie kaktusowym (*cactus stack*). U podstawy jest stos MCP, na poziomie 1 — kod wspólny dla wielu zadań, a z niego wyrastają gałęzie: stosy poszczególnych zadań. Limit wynosił 1024 stosy. Wspólny kod na poziomie 1 oznacza, że jeden egzemplarz programu może obsługiwać wiele zadań jednocześnie — kod jest współużywalny (*re-entrant*). Zadanie może też „pączkować” (*bud*): rozdzielić się na dwa niezależne zadania.

Ostatni szczegół z artykułu jest elegancki: przerwania obsługiwane są jak wejście do procedury.



Rysunek 3. *Stos kaktusowy B6500: z pnia z kodem MCP i kodem wspólnym wyrastają stosy zadań, a rejestry Display pokazują, które procedury są w zasięgu. Zasady zasięgu ALGOL-u przeszły tu ze specyfikacji języka do sprzętu.*

B6700 w oczach użytkowników: Datapro 1973

Raport Datapro z grudnia 1973 roku to najlepszy znany mi obraz linii w szczycie. Typowy B6748 kosztował około 1 237 000 dolarów. B6700 mógł mieć od jednego do trzech procesorów (B6748 — jeden), B7700 — od jednego do siedmiu, przy czym procesorów centralnych i procesorów wejścia-wyjścia razem nie mogło być więcej niż osiem. Drugi procesor dawał według Datapro 70–80 procent przyrostu przepustowości, trzeci — już tylko 30–40 procent.

MCP realizował pamięć wirtualną przez segmentację logiczną. Datapro odnotowuje dwa narzędzia, które przetrwały do dziś: **WFL** (*Work Flow Language*), rozszerzenie sterowania zadaniami, w którym zadania tworzą sieci powiązanych kroków, oraz **CANDE** (*Command and Edit*), gotowy system obsługi terminali. Do tego DMS, system zarządzania danymi. ESPOL udostępniono użytkownikom, głównie do pisania tzw. *installation intrinsics* — procedur instalacji wiązanych z MCP. Dostępne języki: COBOL, FORTRAN, ALGOL, PL/I, APL, BASIC i ESPOL.

Oceny użytkowników były wysokie. Średni czas pracy wynosił według nich 96 procent, a kompilatory i system zebrały *„especially high praise”*. Datapro zestawiał B6748 z IBM 370/145: w tej samej cenie miał dawać od 25 do 100 procent więcej mocy procesora. Dwuprocesorowy B7728 miał być porównywalny z IBM 370/168 za znacznie mniejsze pieniądze.

Trzy rodziny pod jedną nazwą

MCP to nie jeden system

W tej historii łatwo przeoczyć rzecz zasadniczą: nazwy „Master Control Program” Burroughs używał w co najmniej trzech rodzinach maszyn, które nie były ze sobą zgodne ani sprzętowo, ani programowo. Kto czyta stare raporty o „Burroughsie z MCP”, musi najpierw sprawdzić, o którą rodzinę chodzi.

Rodzina	Maszyny	Cechy	System	Los
Large Systems (duże)	B5000, B5500, B5700; B6500–B6900, B7700–B7900; B5900; A Series	stos, deskryptory, znaczniki, ALGOL jako język systemu	MCP w ESPOL, od lat 80. w NEWP	przetrwała jako ClearPath MCP
Medium Systems (średnie)	B2500, B3500, potem B2700–B4700, B2900–B4900	arytmetyka dziesiętna, słowo 16-bitowe, brak arytmetyki binarnej	MCP; dla małych konfiguracji taśmowych BCP	bez następcy
Small Systems (małe)	B1700, B1800, B1900	mikroprogramowane, bez wbudowanego języka maszynowego	MCP I i MCP II w SDL	bez następcy

Medium Systems: maszyny, które nie umiały liczyć binarnie

Raport Datapro z lutego 1970 roku opisuje **B2500 i B3500**, ogłoszone w marcu 1966 roku, z pierwszymi dostawami w maju 1967 roku. To maszyny zupełnie inne niż B5500. Słowo miało 16 bitów (plus bit parzystości) i mieściło dwa bajty 8-bitowe albo cztery cyfry dziesiętne w kodzie BCD. Operandy miały od 1 do 100 cyfr albo bajtów, znaki można było kodować w EBCDIC albo USASCII, a maszyna **nie miała arytmetyki binarnej** — liczyła wyłącznie dziesiętnie.

Datapro nazywa jej system *„Master Control Program (MCP)”* i opisuje go jako *„an integrated operating system”*; dla małych konfiguracji z taśmami przewidziano prostszy BCP (*Basic Control*

Program). Ceny sięgały od około 4400 dolarów miesięcznie (213 000 dolarów zakupu) do 22 200 dolarów miesięcznie (1 060 000 dolarów). Datapro przytacza przykład instalacji, w której jeden operator B3500 wykonał ponad 200 zadań w ciągu ośmiogodzinnej zmiany.

Small Systems: B1700 i maszyna bez języka maszynowego

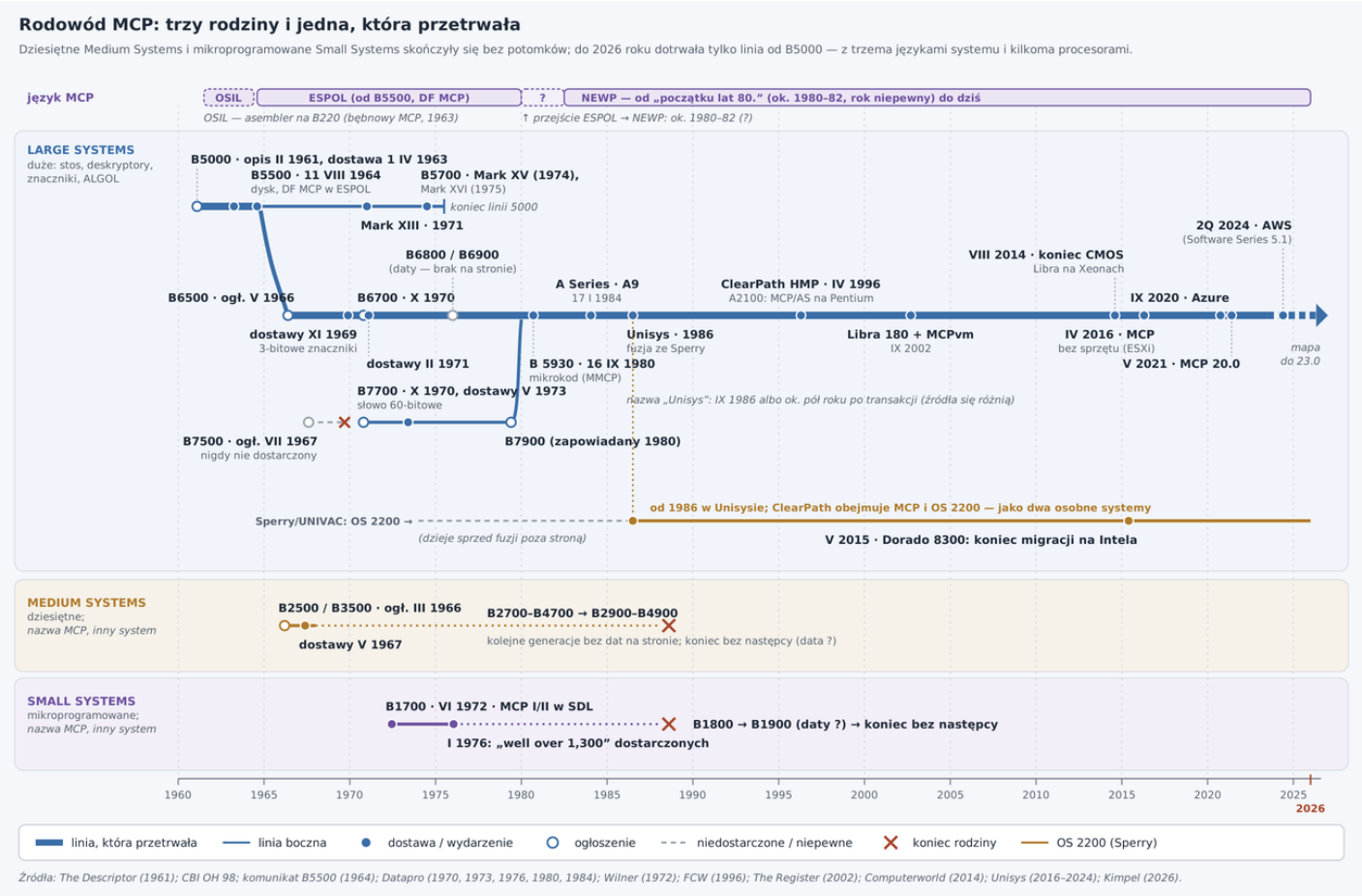
Najdziwniejszą maszyną z nazwą MCP był **B1700**. W. T. Wilner z ośrodka Burroughsa w Golecie w Kalifornii opisał go w artykule *„Design of the Burroughs B1700”* na konferencji FJCC 1972 (s. 489–497). Pierwsze zdanie jest celowo przesadne: *„Burroughs B1700 is a protean attempt to completely vanquish procrustean structures”* — B1700 to proteuszowa próba całkowitego pokonania prokrustowych struktur. Za grą słów stoi konkretna konstrukcja: *„No machine language is built into the hardware”* — w sprzęt nie wbudowano żadnego języka maszynowego.

Zamiast tego B1700 wykonywał mikrokod z pamięci głównej. Każdy język miał własny zestaw rozkazów — Wilner nazywa go „S-językiem” — i własną „S-maszynę”, czyli interpreter w mikrokodzie. Procesory języków COBOL, FORTRAN, BASIC i RPG zmieściły się w mniej niż 4000 mikroinstrukcji 16-bitowych. Sprzęt był *„optimized for interpretation”*, zoptymalizowany pod interpretację, i miał *„emulate existing and future S-machines”* — emulować istniejące i przyszłe S-maszyny. Całą pamięć adresowano co do bitu: *„bytes and words do not exist”*, bajty i słowa nie istnieją.

MCP na B1700 zapewniał pamięć wirtualną, wieloprogramowość i wieloprzetwarzanie, a według Wilnera *„must be written in higher-level language and interpreted just like any other program”* — musiał być napisany w języku wysokiego poziomu i interpretowany jak każdy inny program. Raport Datapro z sierpnia 1976 roku precyzuje: **MCP II** napisano w **SDL** (*Software Development Language*); wymagał co najmniej 32 KB pamięci, a rezydentnie zajmował około 24 KB; najnowsze wydanie 5.1 dawało według Burroughsa 10–13 procent więcej przepustowości. Dla najmniejszych maszyn był wsadowy **MCP I**, wymagający minimum 24 KB. Kompilatora PL/I nie było.

Wilner podaje też pomiary producenta: w RPG wykorzystanie pamięci miało być „siedem razy lepsze” niż na IBM System/3 (4,5 KB wobec 32 KB).

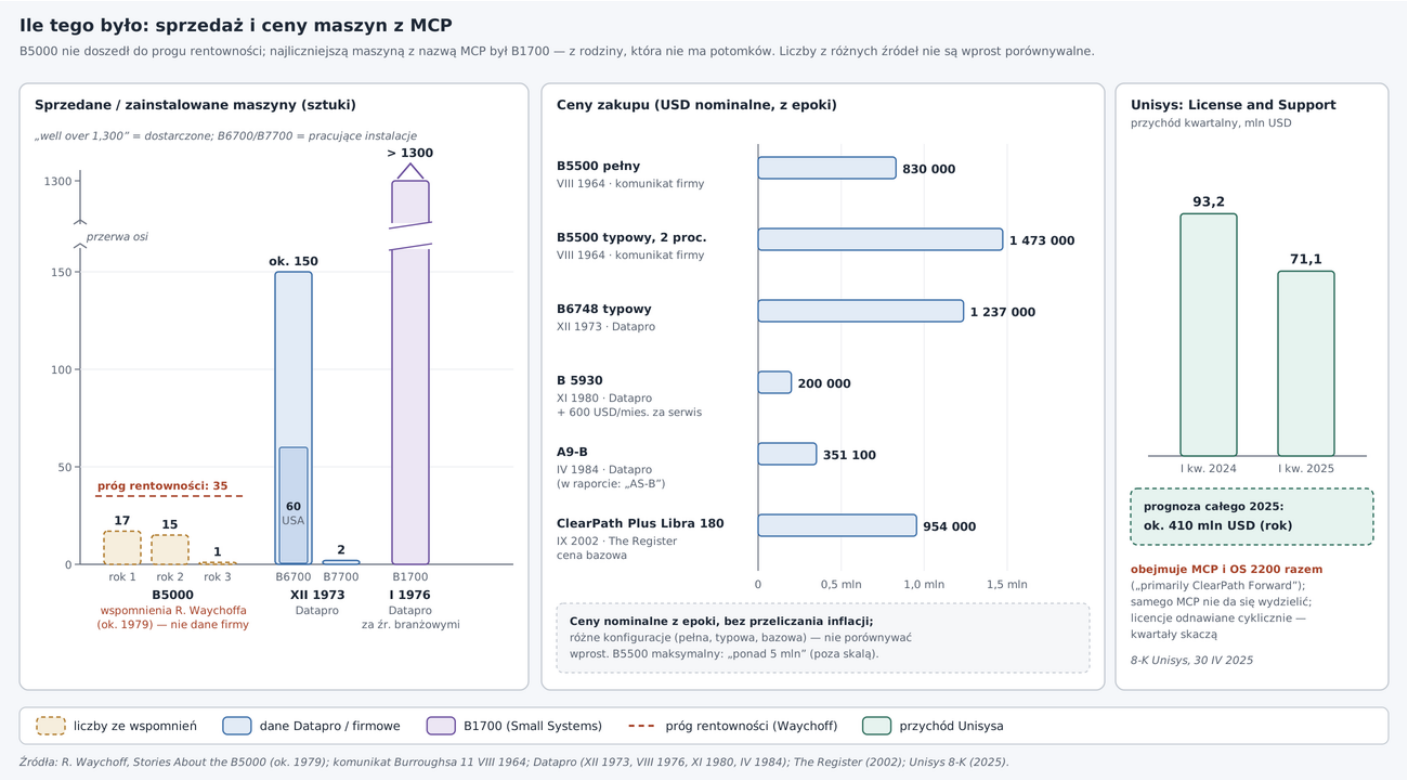
Seria B1700 została ogłoszona w czerwcu 1972 roku. Według Datapro dostawy B1712 i B1714 ruszyły w trzecim kwartale 1972 roku, B1726 — w drugim kwartale 1973 roku, B1728 — w trzecim kwartale 1973 roku. Dzierżawa kosztowała od 1500 do 12 500 dolarów miesięcznie. Do stycznia 1976 roku dostarczono według źródeł branżowych, na które powołuje się Datapro, *„well over 1,300”* systemów — znacznie ponad 1300 — a około 400 czekało w zamówieniach. Dużo — i bez potomków.



Rysunek 4. *Trzy rodziny, jedna nazwa: dziesiętne Medium Systems i mikroprogramowane Small Systems skończyły się bez potomków, a do 2026 roku dotrwała tylko linia od B5000, która po drodze zmieniła trzy razy język systemu i kilka razy procesor.*

Ile tego było: liczby z epoki

Liczby, które padły w tym tekście, pochodzą z bardzo różnych źródeł — od wspomnień Waychoffa po raporty Datapro i komunikaty firmy — i nie da się ich wprost porównywać; ceny są nominalne, z epoki. Zestawia je rysunek.



Rysunek 5. *B5000 nie osiągnął progu rentowności, linia zaczęła się opłacać dopiero z B5500 i B6700, a najliczniejsza maszyna z nazwą MCP — B1700 — należała do rodziny, która nie ma dziś potomków.*

Lata osiemdziesiąte: mikrokod, A Series, NEWP i Unisys

B5900: duży system w mikrokodzie

Na przełomie lat siedemdziesiątych i osiemdziesiątych Burroughs pokazał tańsze wejście do świata swoich dużych systemów: **B 5930**, przedstawiony 16 września 1980 roku. Raport Datapro z listopada 1980 roku opisuje go jako maszynę *„at the entry level to large-scale computing"* — na wejściu do świata dużych komputerów. Najważniejsza zmiana była wewnątrz: architektura wielopoziomowa z procesorem MMCP (*Micro Master Control Processor*), który realizował pełną listę rozkazów dużych systemów Burroughsa **w mikrokodzie**. Logikę zbudowano z układów LSI, pamięć z kości MOS 16K z korekcją błędów, *„without reliance on exotic circuitry"* — bez egzotycznych układów.

B 5930 był w pełni zgodny z B 6900 i z zapowiadanym B 7900. Dawał około 50 procent wydajności B 6900 za połowę ceny, zajmował czwartą część powierzchni i zużywał trzecią część energii.

Kosztował 200 000 dolarów plus 600 dolarów miesięcznie za serwis. Oprogramowanie: MCP, DMS II, NDL II, GEMCOS i CANDE. Datapro nazywa go pierwszą maszyną serii B 5900 i siódmą serią rodziny „900”.

Moim zdaniem mikrokod w B 5930 ma znaczenie dla dalszej historii: lista rozkazów dużych systemów Burroughsa przestała być nierozerwalnie związana z jednym rodzajem logiki. Jeśli dało się ją wykonać przez mikrokod na tańszych układach, to — o czym przekonamy się w rozdziale o ClearPath — można ją było później wykonać także przez warstwę tłumaczącą na zupełnie obcym procesorze.

17 stycznia 1984: A Series

Raport Datapro z kwietnia 1984 roku opisuje nową linię: **A Series**, ogłoszoną **17 stycznia 1984 roku**, z pierwszym modelem A9. Wersja A9-F miała trafiać do klientów od drugiego kwartału 1984 roku, A9-B i A9-D — od pierwszego kwartału 1985 roku. Najważniejsze zdanie raportu mówi o zgodności kodu obiektowego z maszynami B 5000, B 6000 i B 7000. Datapro pisze „B 5000”, ale chodzi zapewne o linię B5900, a nie o maszynę z 1961 roku — to drobna nieścisłość raportu, którą odnotowuję, ale nie rozstrzygam.

Wydajność modeli podano względem B 6900 równego 1,0: A9-B od 0,8 do 1,2, A9-D od 1,3 do 1,7, A9-F od 1,5 do 2,1. Pamięć wynosiła od 6 do 24 MB. Według Burroughsa nowe maszyny potrzebowały o 40–50 procent mniej energii, klimatyzacji i powierzchni niż B 6900. AS-B kosztował 351 100 dolarów, AS-F z dodatkowymi 6 MB pamięci — 613 900. Oprogramowanie obejmowało m.in. MCP, bazę **DMS II**, WFL, sieć BNA i LINC. Konkurentami byli Honeywell DPS 8, IBM 4300 (4381 i 4361) oraz maszyny Sperry 90/80, 1100/60 i 1100/70.

Ta ostatnia linijka ma w sobie ironię: dwa lata później Sperry nie był już konkurentem, tylko częścią tej samej firmy.

ESPOL umiera, rodzi się NEWP

W tym samym czasie zmienił się język, w którym pisano MCP. Paul Kimpel, autor emulatora retro-b5500, napisał 22 czerwca 2026 roku na liście dyskusyjnej cctalk, w wątku o tym, co sprawia, że język komputerowy jest martwy, że ESPOL jest *„dead since the early 1980s”* — martwy od początku lat osiemdziesiątych. Gdy wtedy przepisywano MCP dla linii B6700/B7700, napisano go w nowym języku **NEWP**, który według Kimpela przypomina nieco Modulę. MCP żyje do dziś w Unisys ClearPath MCP, wyewoluowanym właśnie z linii B6x00/B7x00.

Skąd nazwa? Kimpel przytacza dwie opowieści. Według pierwszej NEWP to skrót od *„No Executive Washroom Privileges”* — „bez prawa do łazienki dla kierownictwa”; Kimpel uważa ją za prawdopodobny apokryf. Według drugiej John McClintock, zapytany, czy nowy kompilator jest gotowy, odpowiedział po prostu „newp!” — przekręcone „nope”, czyli „nie”; tę wersję Kimpel uważa za prawdopodobnie prawdziwą. Oficjalnego rozwinięcia nazwy nie znalazłem.

NEWP nie jest ciekawostką muzealną. Stoi na liście kompilatorów w katalogu produktów ClearPath MCP 20.0 z 2021 roku i w karcie ClearPath MCP Developer Studio z 2024 roku. To w nim, nie w ESPOL, jest napisany MCP, który działa dziś w bankach.

Firma: od Macdonalda do Blumenthala

Burroughs jako firma przeżywał w tym czasie własne przemiany. Według kompilacji z encyclopedia.com Ray Macdonald, który w 1964 roku został wiceprezesem wykonawczym, kierował firmą jako prezes w latach 1967–1977. W 1967 roku Burroughs zdobył kontrakt Departamentu Obrony USA na Illiac IV — maszynę, którą dostarczono do NASA Ames w 1972 roku. W 1979 roku na czele stanął W. Michael Blumenthal, a sprzedaż osiągnęła rekordowe 2,8 miliarda dolarów. W lipcu 1980 roku firma odnotowała pierwszy od siedemnastu lat kwartalny spadek zysku. W 1981 roku kupiła System Development Corporation i Memorex.

1986: Burroughs i Sperry, czyli Unisys

Największy ruch przyszedł w połowie dekady. Według tej samej kompilacji w 1985 roku Burroughs zaproponował za akcje Sperry 65 dolarów za sztukę, około 3,7 miliarda dolarów — i spotkał się z odmową. W maju 1986 roku podniósł ofertę do 70 dolarów (około 4,1 miliarda); Sperry bronił się ofertą odkupu własnych akcji po 80 dolarów. Skończyło się na **76,50 dolara za akcję, czyli 4,8 miliarda dolarów** — według encyclopedia.com była to wówczas *„largest merger in computer industry history”*, największa fuzja w historii branży komputerowej. Połączona firma była drugą co do wielkości firmą komputerową w USA, przed DEC.

Nazwę **Unisys** wybrano spośród propozycji pracowników; pochodzi od „United Information Systems”. Źródła różnią się co do tego, kiedy ją przyjęto: computinghistory.org.uk łączy fuzję i nową nazwę z wrześniem 1986 roku, a encyclopedia.com pisze, że nazwę wybrano pół roku po transakcji[5]. W ciągu dwóch lat Unisys zredukował zatrudnienie o 20 procent — 24 000 z 121 000 etatów.

Fuzja połączyła dwa światy, które dotąd ze sobą konkurowały: linię Burroughsa z MCP i linię Sperry/UNIVAC z systemem, który dziś nazywa się OS 2200. Unisys nie połączył ich w jeden system. Do dziś sprzedaje obie — pod wspólną marką ClearPath, ale jako dwa oddzielne systemy. To ważne zastrzeżenie dla każdego, kto szuka śladów MCP: nie każdy „bank na Unisysie” jest bankiem na MCP. Grecki Alpha Bank kupił w 2010 roku ClearPath Dorado 790 — maszynę z linii OS 2200, a nie MCP.

ClearPath: od własnego krzemu do chmury

1990–1996: od Unruha do ClearPath HMP

W 1990 roku następcą Blumenthala został James A. Unruh. W 1991 roku Unisys zawarł ugodę w sprawie zamówień Pentagonu, sięgającą 190 milionów dolarów. W 1996 roku powstała linia **ClearPath**, a w 1997 roku na czele firmy stanął Lawrence A. Weinbach.

Pierwsze kroki ClearPath opisał 14 kwietnia 1996 roku John Moore w „Federal Computer Week” (dziś w archiwum Nextgov). Unisys zapowiedział nowe maszyny A Series: A2800 i A2400, zbudowane w technologii CMOS, oraz **A2100** — z systemem MCP/AS na płycie z procesorem Pentium albo Pentium Pro. Jednocześnie ogłosił linię **ClearPath HMP** (*Heterogeneous Multiprocessing*), opartą na Pentium, Pentium Pro i CMOS, na której miały działać aplikacje MCP/AS, OS 2200, Uniksa i Windows NT. Dostawy miały ruszyć w czerwcu 1996 roku. W moich źródłach to pierwszy przypadek MCP na procesorze, którego nie zaprojektował ani Burroughs, ani Unisys.

2002: Libra 180 i MCPvm

24 września 2002 roku Timothy Prickett Morgan opisał w „The Register” (za ComputerWire) maszynę **ClearPath Plus Libra 180** — pierwszy mainframe po „burroughsowej” stronie Unisysa w architekturze CMP, zbudowany z płyt komórkowych (*cell boards*) takich jak w serwerach ES7000. Mieścił od jednego do ośmiu „burroughsowych” silników CMOS i do 32 procesorów Intela. Natywny MCP działał tylko na silnikach CMOS. Na procesorach Intela działał **MCPvm** — *„virtual machine variant”* MCP, uruchamiany jako aplikacja obok Windows 2000, tańszy i używany głównie do tworzenia oprogramowania.

Cena bazowa wynosiła 954 000 dolarów, a klient mógł dokupować moc na żądanie z opłatą dzienną. Z wydajnością jest kłopot: źródła podają dla tej maszyny 1400 albo 7000 MIPS — obie liczby padają w tym samym artykule[6].

2005–2015: dziesięć lat przenosin na Intela

31 października 2005 roku „Computerworld” opisał zapowiedź Unisysa, którą firma nazwała kierunkową (*directional*): cała linia ClearPath może przejść na procesory Intela. G. Esnouf powiedział wtedy: *„In time, we would plan that ClearPath would run on Intel rather than on the*

current CMOS”. Ta zapowiedź padła w trudnym momencie: Unisys miał w trzecim kwartale 54,3 miliona dolarów straty i zatrudniał 36 000 osób, z których dziesiątą część miał zwolnić.

17 czerwca 2014 roku „Computerworld” donosił, że Unisys wygasza własny procesor CMOS: dla linii Libra (MCP) do końca sierpnia 2014 roku, dla Dorado (OS 2200) do końca 2015 roku; modele 880E i 890E szły do wycofania. Firma pokazała 12 nowych serwerów na procesorach Xeon E5-2600 v2. Oba systemy działają na nich *„through a firmware layer that translates the OS code for execution on x86 chips”* — przez warstwę oprogramowania układowego, która tłumaczy kod systemu tak, by mógł się wykonać na procesorach x86. B. Herkalo z Unisysa uzasadniał to krótko: *„x86 processors are faster and more power efficient”*.

20 maja 2015 roku eWeek opisał Dorado 8300 jako ostatni krok przejścia ClearPath na architekturę Intela — wysiłku trwającego ponad dziesięć lat. Główny inżynier Jim Thompson mówił: *„The ClearPath architectural transition [to Intel] is both a singular achievement and a new beginning”*.

Czy to emulacja?

Legenda mówi, że MCP jest dziś „emulowany na Xeonach”. Źródła są ostrożniejsze. W 2014 roku Unisys mówił o *warstwie oprogramowania układowego tłumaczącej kod systemu*, w 2002 roku — o *wariancie maszyny wirtualnej*. Słowo „emulacja” w żadnym z tych źródeł nie pada. Różnica ma znaczenie praktyczne: to nie jest hobbystyczny emulator w rodzaju Herculesa dla mainframe'ów IBM (o nim pisze strona **Mainframe i z/OS**), tylko produkt producenta systemu, sprzedawany z jego wsparciem. Tutaj piszę więc o „warstwie translacji” albo o „maszynie wirtualnej”, a nie o emulacji.

2016: MCP bez własnego sprzętu

27 kwietnia 2016 roku Unisys ogłosił, że po raz pierwszy oferuje środowisko ClearPath MCP osobno od sprzętu ClearPath Forward. Nowa wersja działała na *„any Intel x86 server under VMware ESXi”* — na dowolnym serwerze x86 z VMware ESXi — albo na systemie spełniającym specyfikację referencyjną. Były dwie edycje: **Bronze**, licencjonowana według liczby jednocześnie pracujących użytkowników, i **Silver**, licencjonowana według pojemności. Dyrektor techniczny Tarek El-Sadany mówił: *„Virtualizing the MCP environment is a major advance...”*.

Miesiąc wcześniej, 30 marca 2016 roku, Unisys wypuścił **MCP Express** i OS 2200 Express — wersje środowisk ClearPath Forward na zwykły pecet z Windows 7 lub nowszym. Wracam do nich w rozdziale o praktyce.

Chmura: Azure i AWS

30 września 2020 roku Unisys uruchomił **ClearPath MCP Software Series for Microsoft Azure**. MCP miał działać *„directly on Azure”*, w chmurze publicznej, bez przepisywania kodu aplikacji. Przy

ogłoszeniu wypowiadali się Vishal Gupta, dyrektor techniczny Unisysa, i Bob Ellsworth z Microsoftu, odpowiedzialny za transformację mainframe'ów.

Później przyszedł AWS. Strona Unisysa bez daty, z okolic 2024–2025 roku, ogłasza: *„ClearPath MCP ecosystems can now run on Amazon Web Services (AWS)”* i dodaje, że wsparcie dla OS 2200 zostanie dodane w połowie 2025 roku. Mapa drogowa z 2023 roku przewidywała natywną obsługę AWS we wszystkich edycjach w wersji Software Series 5.1, w drugim kwartale 2024 roku.

Karta produktu „ClearPath MCP Software Series” z 2024 roku opisuje go jako *„a complete, software-based version of the ClearPath MCP operating environment”* — kompletną, czysto programową wersję środowiska MCP — działającą na VMware i Hyper-V, w chmurze prywatnej i hybrydowej, w edycjach Platinum, Gold, Bronze, Silver, Developer Studio i Developer Studio personal edition. Obietnica brzmi: *„You won't have to rebuild or recompile your applications to accommodate this shift”* — nie trzeba będzie przebudowywać ani rekompilować aplikacji. To zdanie z materiału marketingowego, ale opisuje rzecz, którą linia od B5000 rzeczywiście zrobiła kilka razy. Osobna karta „ClearPath MCP” z tego samego roku wymienia też Pythona, dostępnego w MCP przez *ClearPath Extension Kit for MCP*.

MCP 20.0 i mapa drogowa do 23.0

Najnowszy dokument z pełną listą oprogramowania, jaki znalazłem, to *ClearPath MCP Software Product Catalog, ClearPath MCP 20.0* z maja 2021 roku (dokument 8807 7144-024). Wydanie **MCP 20.0** istnieje więc od 2021 roku. Z wymienionych w nim nowości wybieram te, które pokazują, że to żywy system, a nie relikt:

- w WFL — opcja REPORT w instrukcjach WRAP i UNWRAP;
- w CANDE — komunikat ostrzegawczy, gdy *„the CANDE TANKFILE nears maximum capacity”*, czyli gdy plik roboczy CANDE zbliża się do maksymalnego rozmiaru;
- w Client Access Services — obsługa SMB 3.0;
- szyfrowanie dysków (*Disk Encryption*) dla *„All supported ClearPath Libra and Financial Series systems”* i dla ClearPath MCP Software Series od wydania 3.0.

Katalog wymienia w spisie treści kompilator NEWP i dwa modele licencji: *Base-plus-Usage* (opłata podstawowa plus za zużycie) i *Prepaid Performance* (wydajność opłacona z góry).

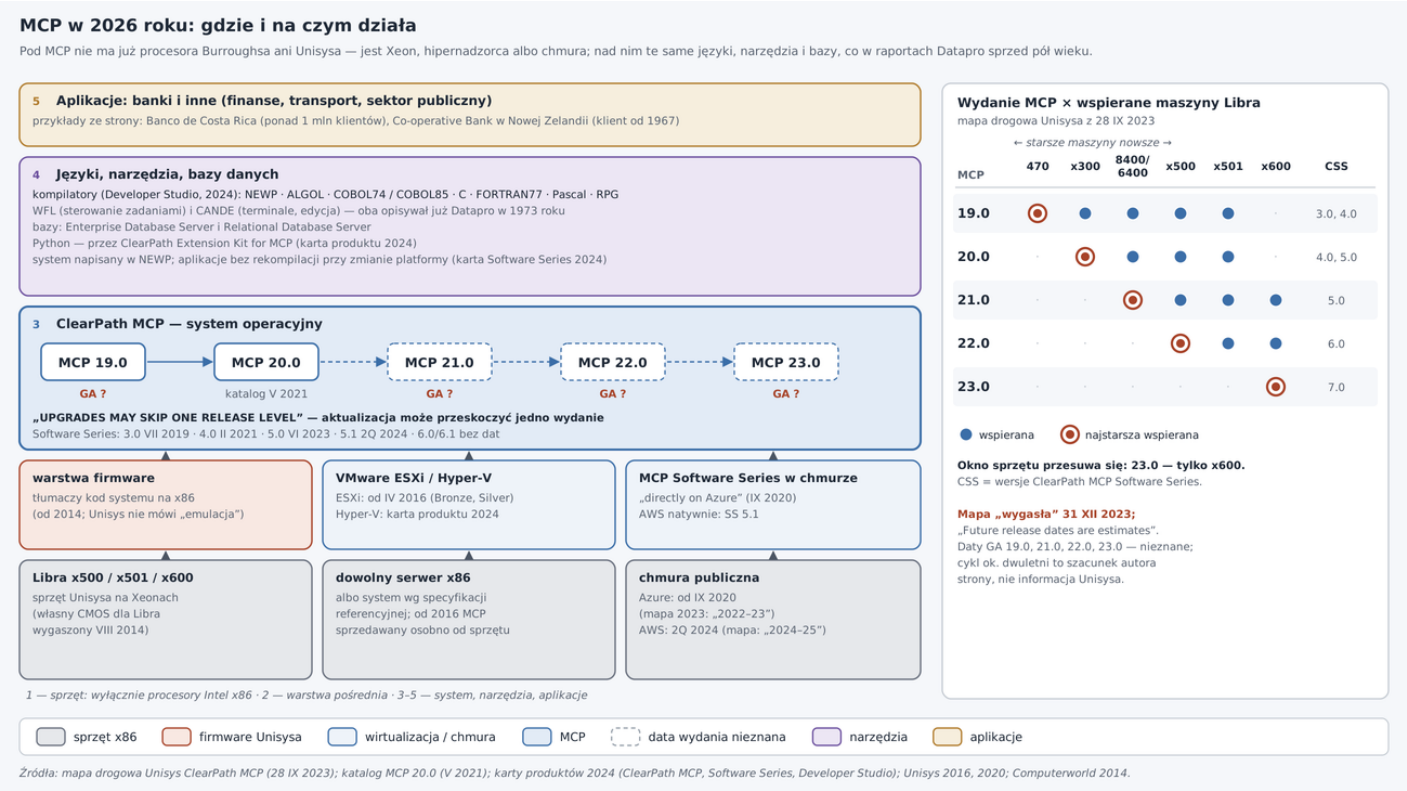
Dalszą przyszłość pokazuje mapa drogowa Unisysa „ClearPath MCP” z 28 września 2023 roku, opatrzona zastrzeżeniem: *„Roadmap Subject to Change. Updated September 27, 2023. Expires December 31, 2023”* i *„Future release dates are estimates”*. Wymienia wydania MCP od 19.0 do 23.0 i maszyny, na których każde z nich będzie działać:

Wydanie MCP	Wspierane maszyny Libra	Wersje Software Series (CSS)	Wybrane tematy
19.0	x501, x500, 8400/6400, x300, 470	3.0, 4.0	—

Wydanie MCP	Wspierane maszyny Libra	Wersje Software Series (CSS)	Wybrane tematy
20.0	x501, x500, 8400/6400, x300	4.0, 5.0	katalog z V 2021
21.0	x600, x501, x500, 8400/6400	5.0	reorganizacja dużych baz, weryfikacja plików kodu, IPsec IKEv2, TLS 1.2, Windows 2022, WebSphere MQ 9.2/9.3
22.0	x600, x501, x500	6.0	TLS 1.3, podpisywanie kodu produktów Unisysa, kryptografia postkwantowa („Crypto Agility”), SMB dla wielu użytkowników
23.0	x600	7.0	—

Na mapie stoi też zdanie wielkimi literami: *„UPGRADES MAY SKIP ONE RELEASE LEVEL”* — przy aktualizacji można pominąć jedno wydanie. Wersje Software Series miały daty: 3.0 — lipiec 2019, 4.0 — luty 2021, 5.0 — czerwiec 2023, 5.1 — drugi kwartał 2024 (natywnie AWS we wszystkich edycjach); 6.0 i 6.1 — bez dat. Chmurę publiczną zaplanowano tak: Azure w latach 2022–2023, AWS w latach 2024–2025. Sprzętowo mapa zapowiada odświeżenie maszyn Libra z hasłem *„Targeting 50% capacity growth”*.

Dat ogólnej dostępności wydań 21.0, 22.0 i 23.0 nie znalazłem. Z mapy wynika tylko ich kolejność, a z dostępnych dat — cykl mniej więcej dwuletni. To mój szacunek, nie informacja Unisysa.



Rysunek 6. *Pod MCP nie ma już własnego procesora Burroughsa ani Unisysa: jest Xeon, hipernadzorca albo chmura. Nad nim — te same języki, narzędzia i bazy, które opisywał Datapro w latach siedemdziesiątych i osiemdziesiątych.*

Bezpieczeństwo: co dawały znaczniki, a co kompilatory

Co robił sprzęt

Mechanizmy sprzętowe są dobrze udokumentowane. W B5000 bit flagi odróżniał kod od danych; Mayer pisze, że sprzęt *„would not execute data or alter code without first having explicitly changed the flag”* — nie wykona danych ani nie zmieni kodu bez jawnej zmiany flagi. Deskryptor z rozmiarem dawał automatyczne sprawdzanie granic. W B6500 wykonanie danych jako kodu albo modyfikacja kodu kończyły się przerwaniem, a stos każdego zadania pilnowały granice BOS i SL. Adam Shostack przytacza komentarz A. Steingruebla, według którego podręcznik B5500 opisuje przerwania Stack Overflow, Invalid Address i Invalid Index: przerywają one zadanie, a system działa dalej.

Ale w dokumentach z epoki nikt nie opisuje tego jako obrony przed atakiem. Shostack w 2008 roku (z aktualizacją z 21 maja 2026 roku) szukał w podręcznikach Burroughsa opisu ataku przez przepełnienie bufora i nie znalazł. Jeden z komentatorów napisał, że nie ma tam *„nothing like the Aleph One paper”*, nic na kształt słynnego tekstu o rozbijaniu stosu; terminem epoki było *„array bounds checking”* — sprawdzanie granic tablic.

Zaufane kompilatory

Levy wskazuje, gdzie w istocie leżała ochrona. Programy użytkowe nie miały bezpośredniej kontroli nad PRT ani nad deskryptorami, ale PRT nie była tylko do odczytu, a system *„relied heavily on the compilers for proper system operation”* — w dużej mierze polegał na kompilatorach. Z jego opisu wynika, że sprzęt zakładał, iż deskryptory i kod wytworzyły kompilator albo MCP. Jeśli więc ktoś zdołał uruchomić kod, którego nie wytworzył zaufany kompilator, cała ochrona przestawała obowiązywać.

Test penetracyjny z 1981 roku

Tę słabość wykorzystali A. L. Wilkinson i współautorzy w pracy *„A Penetration Analysis of a Burroughs Large System”*, opublikowanej w styczniu 1981 roku w „Operating Systems Review”

(tom 15, nr 1, s. 14–25). Badali B6700. Typ pliku „code-file”, oznaczający plik wykonywalny, mógł nadać tylko zaufany kompilator. Okazało się jednak, że typ pliku zapisanego na taśmie przechowywany jest w etykiecie taśmy, a ta etykieta *„is not integrity checked”* — nie jest chroniona przed zmianą. Kto miał dostęp do taśmy poza systemem, mógł więc zmienić oznaczenie zwykłego pliku na plik wykonywalny i wczytać go z powrotem — a taki program miał dostęp do dowolnej pamięci. Zalecenie autorów było proste: nie ładować plików wykonywalnych z taśmy. Opis tej podatności przechowuje baza podatności Uniwersytetu Kalifornijskiego w Davis, opracowana przez Matta Bishopa w 1999 roku.

Wniosek jest taki sam jak na stronie **IBM i: System/38 i AS/400**: znaczniki w pamięci są akceleratorem, a strażnikiem jest łańcuch zaufanego kodu — od kompilatora przez system plików po nośniki. Wystarczy przerwać ten łańcuch w jednym miejscu, a sprzęt posłusznie wykona to, co dostanie.

Hasło reklamowe i raport roczny

Karta produktu ClearPath MCP z 2024 roku nazywa go *„Most secure operating system, where client data has never been breached”* — najbezpieczniejszym systemem, w którym dane klientów nigdy nie wyciekły. To hasło marketingowe, którego nie umiem zweryfikować. Warto zestawić je ostrożnie z raportem rocznym Unisysa (formularz 10-K za 2025 rok), który przyznaje, że cyberatak z 2022 roku skończył się wykradzeniem kodu źródłowego oprogramowania produktów i platform firmy — bez wskazania, czy chodziło o MCP. Kod źródłowy producenta to nie dane klientów, więc oba zdania mogą być jednocześnie prawdziwe. Ale „nigdy” w reklamie i „wykradziono” w raporcie dla inwestorów stoją obok siebie.

Ślady w cudzych pomysłach: Kay, Smalltalk, kod bajtowy

Najlepiej udokumentowany wpływ B5000 opisał sam zainteresowany. Alan Kay w eseju *„The Early History of Smalltalk”* (1993) wspomina służbę w Air Training Command amerykańskiego lotnictwa około 1961 roku. Format plików na Burroughs 220 był dla niego pierwszym ledwo uchwyconym przeczuciem idei obiektów — *„Needless to say, this was a great idea, and was used in many subsequent systems”*. Drugie przyszło wkrótce: *„The second barely-seeing of the idea came just a little later when ATC decided to replace the 220 with a B5000”*. Kay przyznaje: *„I didn't have the perspective to really appreciate it at the time, but I did take note of its segmented storage system”* — wtedy nie umiał tego docenić, ale zapamiętał segmentowaną pamięć. Dodaje jednak, że *„my big hit from this machine at this time was not the OOP idea”* — z tej maszyny wyniósł wtedy coś innego niż programowanie obiektowe: sposób tłumaczenia i wartościowania języków wysokiego poziomu.

Później, gdy Barton był profesorem w Utah, Kay usłyszał od niego zdanie: *„The basic principle of recursive design is to make the parts have the same power as the whole”* — zasadą projektowania rekurencyjnego jest dać częściom tę samą moc co całości. To skłoniło go do myślenia o komputerze jako jednostce projektu, a dalej — przez Simulę — o obiektach. W maszynie FLEX *„Object references were handled … as a generalization of B5000 descriptors”* — referencje do obiektów były uogólnieniem deskryptorów B5000. Kay zastrzega, że *„The B5000 just didn't directly scale down into a tiny machine”*, ale jego kod bajtowy — owszem. W dedykacji eseju wymienia m.in. Boba Bartona.

Alan Kay z prototypem Dynabooka, 5 XI 2008 roku

Alan Kay z prototypem Dynabooka, 5 XI 2008 roku. Źródło: Flickr, CC BY 2.0, autor: Marcin Wichary.

Douglas W. Jones w notatkach z 2004 roku porównuje kod B5000 z kodem bajtowym Javy i nazywa je *„identical in spirit”* — identycznymi w duchu, dodając, że bezpieczeństwo B5000 było słabsze niż Javy i poprawiono je w późniejszych modelach. To opinia wykładowcy, nie dokument wpływu. Bezpośredniego wpływu B5000 na Intel iAPX 432 czy na projektantów maszyny wirtualnej Javy nie znalazłem w źródłach pierwotnych; jedyny udokumentowany łańcuch prowadzi przez Kaya.

MCP w 2026 roku: kto płaci i za co

W raporcie rocznym za 2025 rok Unisys opisuje ClearPath Forward jako *„A flexible collection of Unisys proprietary core software operating systems, products, and platforms”*. Segment ECS, do którego należy, obsługuje według firmy *„leading entities in financial services, travel and transportation”*, a także ochronę zdrowia i sektor publiczny — głównie władze stanowe i lokalne w USA oraz rządy poza USA. Licencje terminowe na systemy ClearPath Forward Unisys księguje w całości przy odnowieniu umowy, a kontrakty mają limity zużycia. 31 grudnia 2025 roku Unisys zatrudniał około 15 000 osób: 4200 w Indiach, 2300 w USA i 8500 w innych krajach.

Ile to pieniędzy? W wynikach za pierwszy kwartał 2025 roku (8-K z 30 kwietnia 2025 roku) przychód z licencji i wsparcia (*License and Support*, L&S) wyniósł 71,1 miliona dolarów wobec 93,2 miliona rok wcześniej, a prognoza na cały 2025 rok zakładała około 410 milionów. L&S to według Unisysa *„primarily ClearPath Forward”* — obejmuje więc łącznie MCP i OS 2200 i nie da się z niego wydzielić samego MCP. Ponieważ licencje odnawia się cyklicznie, kwartały skaczą.

Kto konkretnie płaci za MCP? Udokumentowanych przykładów jest niewiele. **Banco de Costa Rica** w studium przypadku Unisysa (opublikowanym w 2021 roku, zmienionym 12 maja 2023 roku) korzysta z ClearPath MCP, w tym z MCP Developer Studio, oraz z ClearPath ePortal i pamięci masowej Dell EMC; bank deklaruje ponad milion klientów i ponad 250 oddziałów, a jego administrator baz danych Luis Ramirez mówi o partnerstwie trwającym *„over two decades”*. **Co-operative Bank** z Nowej Zelandii wymienił w czerwcu 2016 roku maszyny ClearPath Libra 690 i ES7000 na ClearPath Forward 4390, na którym MCP działa obok Windows i VMware; Unisys jest partnerem banku od 1967 roku, a jego system bankowy NTBS działa od 2008 roku. Linie lotnicze w

moich źródłach wiążą się z OS 2200, a nie z MCP.

Praktyka: jak dotknąć MCP w 2026 roku

Emulator B5500 w przeglądarce

Najprostsza droga to **retro-B5500** — emulator Burroughsa B5500 napisany w JavaScripcie i działający w przeglądarce, z prawami autorskimi „2013, Nigel Williams and Paul Kimpel” i na licencji MIT. Projekt zaczął się na Google Code, w czerwcu 2015 roku przeniósł się na GitHub (pkimpel/retro-b5500). Na blogu Paul Kimpel z San Diego opisuje jego dzieje: pierwsze wpisy z marca 2012 roku, projektowanie emulatora „late 2012 and early 2013”, i to, że Nigel Williams *„more or less insisted”*, by pisać go w JavaScripcie dla przeglądarki. Wydanie 1.05 ukazało się w marcu 2018 roku, 1.06 — 3 stycznia 2020 roku (ogłoszone 15 lutego 2020 roku). W lutym 2022 roku Jim Fehlinger, który przepisał kilka dużych listingów źródeł B5500, uruchomił emulator w Electronie, bez przeglądarki, także na Raspberry Pi 4. Drugim celem projektu jest rekonstrukcja źródeł i kodu obiektowego systemu.

Oprogramowanie systemowe to kompletne wydanie **Mark XIII z 1971 roku** — taśmy i pliki ASCII, udostępniane *„under liberal terms of a Unisys educational/hobbyist license”*, na liberalnych warunkach licencji edukacyjnej i hobbystycznej Unisysa, po złożeniu wniosku przez stronę Software Request. Do inicjalizacji trzeba pobrać co najmniej obraz taśmy SYSTEM. Osobno dostępna jest biblioteka programów użytkowników CUBE w wersji 13 z 1971 roku.

Kolejność kroków jest taka: najpierw strona projektu (phkimpel.us/B5500, ostatnia rewizja z 19 lipca 2026 roku), potem przewodnik *Getting Started* w wiki na GitHubie, wniosek o taśmy, a na końcu konsola startowa `B5500Console.html`. Wiki opisuje m.in. konfigurację podziału czasu CANDE, pracę ze stanowiskiem operatora SPO, czytnik kart, drukarkę, taśmy i transmisję danych. Poleceń konsoli tutaj nie przepisuję — wiki projektu jest jedynym miejscem, z którego warto je brać.

Współczesny MCP: Express i Developer Studio

30 marca 2016 roku Unisys wypuścił **MCP Express** i OS 2200 Express — wersje środowisk ClearPath Forward na pecet z Windows 7 lub nowszym, *„available immediately at no cost”*, dostępne od razu bezpłatnie, ale *„Each is subject to an annual recurring license”* — każda wymaga licencji odnawianej co roku. Adresatami byli programiści aplikacji korporacyjnych i środowisko

akademickie.

Jak to wyglądało w praktyce, opisali hobbyści na liście cctalk w czerwcu i lipcu 2020 roku: wniosek przez formularz, link do pobrania następnego dnia, pakiet o rozmiarze 2,66 GB. Licencja *„expires on July 31st of each calendar year and must be renewed by re-registering with Unisys and then downloading and installing a new package”* — wygasa 31 lipca każdego roku i trzeba ją odnowić, rejestrując się ponownie i instalując nowy pakiet. Obraz dysku i konfiguracja maszyny są przechowywane osobno. A. Perry uruchomił całość na Windows 10 w maszynie wirtualnej na MacBooku — po raz pierwszy od 1989 roku, kiedy ostatnio używał MCP.

Czy MCP Express jest nadal oferowany w 2026 roku, nie potwierdziłem. Karta produktu z 2024 roku wymienia za to **ClearPath MCP Developer Studio**, pakiet do tworzenia aplikacji z kompilatorami ALGOL, C, COBOL74, COBOL85, FORTRAN77, NEWP, Pascal, RPG, SORT i Data Comm ALGOL, z CANDE i bazami Enterprise Database Server i Relational Database Server. Ma edycje Platinum i Team (bez limitu programistów), Standard (od 5 do 25) i Personal (jeden), tę ostatnią na pecet albo serwer x86; Team i Standard działają także w Azure. Ceny karta nie podaje, a licencje są imienne albo pojemnościowe — Developer Studio nie jest opisany jako darmowy. Produktu o nazwie „MCP Software Developer's Kit” nie znalazłem.

Co czytać

Do samodzielnej lektury najlepsze są skany w bitsavers: biuletyn „The Descriptor” z 1961 roku, podręcznik ESPOL z 1967 roku, opisy B5500 MCP i książka Organicka z 1973 roku, a także pełen katalog raportów Datapro o Burroughsie — od B2500/B3500 z 1970 roku po A Series. Ten sam autor, który stworzył retro-b5500, udostępnia też emulatory Datatrona 205, Burroughsa 220, Bendiksa G-15, IBM 1620 i LGP-21.

Mity i sprostowania

Człon opowieści	Werdykt	Dlaczego
„B5000 nie miał w ogóle asemblera”	fałsz dla twórców systemu	Bębnowy MCP pisano w OSIL, asemblerze działającym na B220; zespół trzymał to w tajemnicy (CBI OH 98, s. 38).
„MCP zawsze pisano w ESPOL”	fałsz	Najpierw OSIL; ESPOL powstał z Extended ALGOL w około trzy tygodnie; od początku lat 80. MCP pisany jest w NEWP.
„Pierwsza pamięć wirtualna na świecie”	fałsz	Atlas działał od XII 1962, przed dostawą B5000. Dla B5000 — najwyżej „być może pierwszy komercyjny”, z segmentacją.

Człon opowieści	Werdykt	Dlaczego
„B5000 miał trzybitowe znaczniki”	fałsz	B5000 — jeden bit flagi; 3 bity znacznika w słowie 51-bitowym ma dopiero B6500.
„Znaczniki sprzętowo uniemożliwiały włamanie”	fałsz	Ochrona opierała się na zaufanych kompilatorach; w 1981 roku opisano penetrację B6700 przez podrobiony typ pliku na taśmie.
„Projektowano go jako obronę przed przepełnieniem bufora”	anachronizm	Dokumenty epoki mówią o sprawdzaniu granic tablic i przerwaniach Invalid Index, nie o atakach.
„B5000 był sukcesem”	fałsz na początku	Według Waychoffa 17, 15 i 1 maszyna w trzech latach przy progu 35; „Nothing worked”. Sukces przyszedł z B5500 i B6700.
„MCP to jeden system”	fałsz	Trzy niezgodne rodziny: Large, Medium (dziesiętne) i Small (B1700, MCP I/II w SDL).
„Nazwa MCP pochodzi z Trona”	fałsz	Nazwa stoi w biuletynie Burroughsa z II 1961; związek w drugą stronę — niepotwierdzony.
„B5000 był mikroprogramowany”	bez potwierdzenia	Źródła pierwotne o B5000 tego nie mówią; mikroprogramowane były B1700 (1972) i MMCP w B5900 (1980).
„B7500 był kolejną maszyną linii”	fałsz	Ogłoszony VII 1967, nigdy nie dostarczony; zastąpił go B7700.
„Barton był szefem projektu”	fałsz	Sam nazwał się „sabotażystą przemysłowym”; podpisał artykuł jako konsultant; zespół pracował w Pasadenie w grupie Lonergana.
„MCP na Xeonach to emulator jak Hercules”	myląco	Unisys mówi o warstwie firmware tłumaczącej kod i o MCPvm; to produkt producenta, nie emulacja hobbystyczna.
„Unisys rozdaje MCP za darmo do nauki”	częściowo, kiedyś	MCP Express (2016) był bezpłatny z roczną licencją; Developer Studio (2024) jest płatny; status Express w 2026 niepotwierdzony.

Epilog

Barton chciał w 1961 roku maszyny, na której nikt nie będzie potrzebował języka maszynowego. Jego zespół zbudował ją, ale pierwszy system napisał w asemblerze innego komputera i starał się to ukryć. Sześćdziesiąt lat później zasada spełniła się w sposób, którego Barton nie mógł przewidzieć: MCP nie ma już własnego procesora, a jego programy nie wiedzą o tym — wykonują je Xeony w VMware albo w chmurze, przez warstwę tłumaczącą listę rozkazów, którą zaprojektowano jako cel kompilatora ALGOL-u.

Z legendy zostaje mniej, niż obiecuje, i więcej, niż się zwykle wydaje. B5000 nie był pierwszy z pamięcią wirtualną, nie był sukcesem handlowym i nie był niezniszczalny. Ale był maszyną, w której język wysokiego poziomu był punktem wyjścia, a nie dodatkiem — i której potomek, przez A Series i ClearPath, wciąż prowadzi rachunki banków.

Ciekawostki zebrane

- Pierwszy B5000 wysłano klientowi w prima aprilis — 1 IV 1963 roku.
- Knuth zaproponował w 1960 roku napisanie kompilatora ALGOL 58 dla B205 w wakacje za 5000 dolarów; Burroughs podejrzewał, że już go ma.
- Dijkstra po kompilacji `BEGIN END.` w niecałe trzy sekundy: „How does one get any conception of how fast it is?”.
- „The Algol Roadmap” wydrukowano w około 11 000 egzemplarzy, gdy programistów na świecie było około 14 000.
- NEWP to według jednej opowieści „No Executive Washroom Privileges”, według drugiej — „newp!” Johna McClintocka.
- B1700 nie miał wbudowanego języka maszynowego; każdy język dostawał własną „S-maszynę” w mikrokodzie.
- Licencja MCP Express wygasała co roku 31 lipca.
- Co-operative Bank z Nowej Zelandii jest klientem Burroughsa i Unisysa od 1967 roku.
- Nazwę „Unisys” zaproponowali pracownicy — od „United Information Systems”.
- Smithsonian (NMAH) przechowuje maszynę sumującą Burroughs Style 9 z lat 1911–1913 i eksperymentalną maszynę sumującą z 1911 roku (Model No. 106), dar Burroughs Corporation.
- Film „Tron” (1982) ma złowrogi program sterujący o nazwie MCP. Scenarzystka Bonnie MacBird wysłała szkic scenariusza Alanowi Kayowi, który poprawiał go na komputerze Alto, a z jego pomysłów miały się wziąć m.in. elementy świata filmu — w tym MCP jako „inna nazwa systemu operacyjnego” (CINEMORE, 2020). Artykuł nie łączy tej nazwy z Burroughsem. Kay pracował wcześniej na B5000, więc łańcuch „od Burroughsa przez Kaya do filmu” jest kuszący — ale to mój domysł, bez źródła, które mówiłoby to wprost.

Oś czasu

Data	Wydarzenie
1885	W. S. Burroughs wynajduje rejestrującą maszynę sumującą
1886	American Arithmometer Company w St. Louis
1888 albo 1892	patent Burroughsa[1]
1898	śmierć W. S. Burroughsa
1904 albo 1905	Detroit, Burroughs Adding Machine Company[2]
VI 1956	zakup ElectroData (Pasadena); Datatron 205 staje się B205
lato 1958	Barton opisuje tłumaczenie wyrażeń wzorowane na notacji Łukasiewicza
I 1959 – IV 1961	grupa Product Planning Lonergana w Pasadenie
IV 1960	Barton: pierwszy bęben „cannot be optional”
lato 1960	Knuth pisze ALGOL 58 dla B205
II 1961	biuletyn „The Descriptor”; nazwa Master Control Program
9–11 V 1961	artykuł Bartona na WJCC; w V 1961 opis w „Datamation”
7 XII 1962	inauguracja Atlasa w Manchesterze
1 IV 1963	pierwsza dostawa B5000 (UTC); drugi klient — NASA
X 1963	pierwsze wydanie COBOL-a dla B5000
11 VIII 1964	ogłoszenie B5500 z dyskiem; dyskowy MCP w ESPOL
III 1966	B2500 i B3500 (Medium Systems); dostawy od V 1967
V 1966	ogłoszenie B6500
VII 1967	ogłoszenie B7500, nigdy niedostarczonego
1967	Ray Macdonald prezesem; kontrakt na Illiac IV
1968	Hauck i Dent opisują stos B6500 na SJCC
XI 1969	pierwsze dostawy B6500
X 1970	B6700 i B7700; dostawy od II 1971 i V 1973
1971	B5500 Mark XIII — wydanie dostępne dziś w emulatorze
VI 1972	seria B1700 (Small Systems), MCP II w SDL
VII 1974, VI 1975	B5700 Mark XV.3.0 i Mark XVI.0
16 IX 1980	B 5930 z mikrokodowanym MMCP
I 1981	Wilkinson i in.: analiza penetracyjna dużego systemu Burroughsa
początek lat 80.	MCP przepisany w NEWP
17 I 1984	A Series (A9)

Data	Wydarzenie
6 IX 1985	Burroughs B 5000 Conference w Marina del Rey
1986	przejęcie Sperry za 4,8 mld dolarów; Unisys[5]
14 IV 1996	ClearPath HMP; A2100 z MCP/AS na Pentium
24 IX 2002	ClearPath Plus Libra 180 i MCPvm
31 X 2005	zapowiedź przejścia ClearPath na Intela
VIII 2014	koniec własnych procesorów CMOS dla Libra
20 V 2015	Dorado 8300 kończy migrację ClearPath na Intela
30 III 2016	MCP Express i OS 2200 Express
27 IV 2016	ClearPath MCP niezależny od sprzętu (VMware ESXi)
30 IX 2020	ClearPath MCP Software Series for Azure
V 2021	katalog MCP 20.0
2022	cyberatak na Unisys, kradzież kodu źródłowego
28 IX 2023	mapa drogowa MCP 19.0–23.0
2Q 2024	Software Series 5.1 — natywnie w AWS
30 IV 2025	wyniki I kwartału: L&S 71,1 mln dolarów
22 VI 2026	Kimpel na cctalk o śmierci ESPOL i narodzinach NEWP

Wątek polski

Notacja Łukasiewicza u źródła stosu

Najmocniejszy polski ślad w tej historii nie jest instalacją ani firmą, tylko pomysłem. Robert Barton w artykule z maja 1961 roku pisze wprost: *„The Polish notation was invented by the logician Lukasiewicz for use in the propositional calculus”* — i projektuje maszynę wokół programu w postaci „polskiego ciągu”. David Dahm na konferencji w 1985 roku datuje zainteresowanie Bartona na lato 1958 roku: Barton opisywał wtedy metodę tłumaczenia wyrażeń inspirowaną notacją, której autora transkrypcja zapisuje jako „Lukaszewicz” (s. 48). Paul King dodał, że notacja polska *„implies a stack”* i że na niej oparto wczesny projekt B5000 (s. 46–50). Logika zdań Jana Łukasiewicza dała więc architekturze dużych systemów Burroughsa — a przez nie dzisiejszemu ClearPath MCP — zasadę, na której stoi stos.

Polskie kalendarium

Zbigniew Ryznar w „Tabeli 50 faktów”, światowym kalendarium 50 firm i organizacji informatycznych (Biblioteka Cyfrowa Politechniki Warszawskiej, około 2011 roku), pisze o Burroughsie, że w 1904 roku zmienił nazwę na Burroughs Adding Machine Company, a w 1986 roku w wyniku fuzji ze Sperry Rand wszedł w skład Unisysa. O B5000 pisze: „W 1961 ukazuje się komputer B5000 — pierwszy komercyjny komputer z segmentowaną pamięcią wirtualną. Zastosowano w nim też mikroprogramową obsługę polskiej notacji.” Drugie zdanie trzeba czytać ostrożnie: źródła pierwotne nie opisują B5000 jako maszyny mikroprogramowanej — mikroprogramowane były dopiero B1700 i procesor MMCP w B5900[7]. To najpewniej uproszczenie.

Unisys Polska

Spółka Unisys Polska istniała co najmniej w 2006 roku: 27 czerwca 2006 roku jej prezesem zarządu został Dariusz Mazurek, wcześniej, od czerwca 2004 roku, dyrektor pionu instytucji finansowych (gsmonline.pl). Artykuł nie mówi nic o polskich klientach ClearPath MCP i takich klientów w źródłach z adresem nie znalazłem.

Źródła i dalsze lektury

Przypisy

[1] Patent z 21 VIII 1888: computinghistory.org.uk, Burroughs Corporation; patent na model nadający się do sprzedaży w 1892: encyclopedia.com, Unisys Corporation.

[2] 1904: computinghistory.org.uk i Z. Ryznar, Tabela 50 faktów; 1905: encyclopedia.com.

[3] Wynalazek pamięci wirtualnej w Atlasie (XII 1962): IEEE Milestone, Atlas Computer and the Invention of Virtual Memory; B5000 „perhaps the first commercial computer to provide virtual memory”: Mayer (1982); „pierwszy komercyjny komputer z segmentowaną pamięcią wirtualną”: Ryznar.

[4] 1961 (opis i ogłoszenie): Burroughs, The Descriptor (II 1961), Barton (V 1961), encyclopedia.com, Ryznar; 1963 (pierwsza dostawa 1 IV 1963): CBI OH 98, s. 26 i 41.

[5] Fuzja i nazwa Unisys w IX 1986: computinghistory.org.uk; nazwa przyjęta pół roku po transakcji: encyclopedia.com.

[6] 1400 MIPS i 7000 MIPS — obie wartości w tym samym tekście: The Register / ComputerWire (24 IX 2002).

[7] „Mikroprogramowa obsługa polskiej notacji” w B5000: Ryznar; opisy B5000 bez mikroprogramowania: Levy, rozdz. 2, Mayer; mikroprogramowany B1700: Wilner (1972); mikrokodowany MMCP w B 5930: Datapro (XI 1980).

Źródła pierwotne i wspomnienia

- Burroughs B 5000 Conference, 6 IX 1985 (CBI OH 98), transkrypcja; rekord w UMN Conservancy
- Richard Waychoff, Stories About the B5000 and People Who Were There
- R. S. Barton, A New Approach to the Functional Design of a Digital Computer (1961); kopia w CHM
- Burroughs, The Descriptor (II 1961); komunikat o B5500 (VIII 1964); B5500 ESPOL Reference Manual (X 1967)
- bitsavers: katalog B5000/5500/5700; katalog Large Systems; E. I. Organick, Computer System Organization (1973)

Architektura

- Henry M. Levy, Capability-Based Computer Systems, rozdz. 2
- A. J. W. Mayer, The Architecture of the Burroughs B5000: 20 Years Later and Still Ahead of the Times? (1982)
- E. A. Hauck, B. A. Dent, Burroughs' B6500/B7500 Stack Mechanism (1968)
- Douglas W. Jones, notatki o Burroughsie (2004)
- W. T. Wilner, Design of the Burroughs B1700 (1972)
- IEEE Milestone: Atlas; Alan Kay, The Early History of Smalltalk (1993)

Raporty Datapro

- B2500/B3500 (II 1970); B6700/B7700 (XII 1973); B1700 (VIII 1976); B 5900 (XI 1980); A9 (IV 1984)
- pełny katalog raportów Datapro o Burroughsie

Firma, Unisys i ClearPath

- computinghistory.org.uk, Burroughs Corporation; encyclopedia.com, Unisys Corporation
- FCW/Nextgov (1996); The Register (2002); Computerworld (2005); Computerworld (2014); eWeek (2015)
- Unisys, ClearPath MCP wirtualny (2016); iTWire (2016); ExecutiveGov, Azure (2020); Unisys, AWS
- katalog ClearPath MCP 20.0 (2021); mapa drogowa MCP (2023); karty z 2024: ClearPath MCP, Software Series, Developer Studio
- Unisys, wyniki I kw. 2025; Unisys, 10-K za 2025
- Banco de Costa Rica; Co-operative Bank (2016); Alpha Bank (2010)

Bezpieczeństwo

- UC Davis, opis analizy Wilkinsona i in. (1981); Adam Shostack, Buffer Overflows and History (2008/2026)

Emulator, Express, NEWP

- retro-B5500, strona projektu; konsola; GitHub; Getting Started; pomoc; blog
- Unisys, MCP Express (2016); cctalk 2020: 1, 2, 3
- Paul Kimpel o ESPOL i NEWP (cctalk, 2026); CINEMORE o scenariuszu „Trona” (2020)

Polskie

- Zbigniew Ryznar, Tabela 50 faktów (Biblioteka Cyfrowa PW); gsmonline.pl, Nowy prezes Unisys Polska (2006)

Ilustracje

Diagramy i animacje są moje. Zdjęcia pochodzą ze zbiorów cyfrowych muzeów, kolekcji i serwisu Flickr, które udostępniają je jako domenę publiczną albo na licencjach Creative Commons; część jest na licencjach niekomercyjnych (NC) lub bez utworów zależnych (ND) — ta wiki jest prywatna i niekomercyjna, obrazy pokazujemy bez zmian. Obrazy są wyświetlane bezpośrednio z serwerów

instytucji i autorów; licencja i autor stoją w każdym podpisie. Materiałów objętych zastrzeżeniem praw nie osadzamy — odsyłamy do nich w źródłach.

Nota metodyczna: czego nie wiemy

- **Liczby sprzedaży B5000** pochodzą tylko ze wspomnień Waychoffa; nie znalazłem źródła firmowego ani łącznej liczby wyprodukowanych B5000 i B5500, ani dziennej daty ogłoszenia B5000.
- **Kulisy projektu** znam z relacji uczestników spisanych po 18–24 latach; gdzie się różnią (np. Barton o bębnach), podaję obie.
- **UTC**, pierwszy klient — rozwinięcie skrótu niepotwierdzone; **OSIL** — rozwinięcia nie znalazłem; tożsamość „Erdwina” — nieznana.
- **NEWP**: oficjalnego rozwinięcia i dokładnego roku przejścia MCP na NEWP nie ustaliłem; obie opowieści o nazwie znam z jednego źródła.
- **Daty ogólnej dostępności** MCP 19.0, 21.0, 22.0 i 23.0 — nieznane; mam tylko katalog 20.0 (V 2021) i mapę z 2023 roku, która sama się zastrzega.
- **MCP Express w 2026** — dostępności nie potwierdziłem; pochodzenia taśm Mark XIII w emulatorze nie badałem.
- **Klienci MCP**: tylko dwa przykłady z adresem (Kostaryka, Nowa Zelandia); przychody Unisysa łączą MCP z OS 2200.
- **Wpływ na Javę i iAPX 432** — bez źródła pierwotnego; jest tylko opinia D. W. Jonesa i świadectwo Kaya.
- **Tron** — związek nazwy z Burroughsem niepotwierdzony.
- **Wątek polski**: żadnej instalacji Burroughsa ani Unisysa z MCP w Polsce nie znalazłem w źródłach z adresem.
- **Typy źródeł**: dokumenty Burroughsa i Unisysa (biuletyny, komunikaty, katalogi, karty marketingowe — te ostatnie oznaczam jako reklamę), raporty Datapro (niezależny wydawca, ale część danych od producenta), artykuły konferencyjne, prasa branżowa, wspomnienia i listy dyskusyjne. Książek Organicka i Levy'ego w całości nie czytałem.

Revision #6
Created 2026-10-26 12:53:40 UTC by Tomasz Konieczny
Updated 2026-11-01 17:50:18 UTC by Tomasz Konieczny