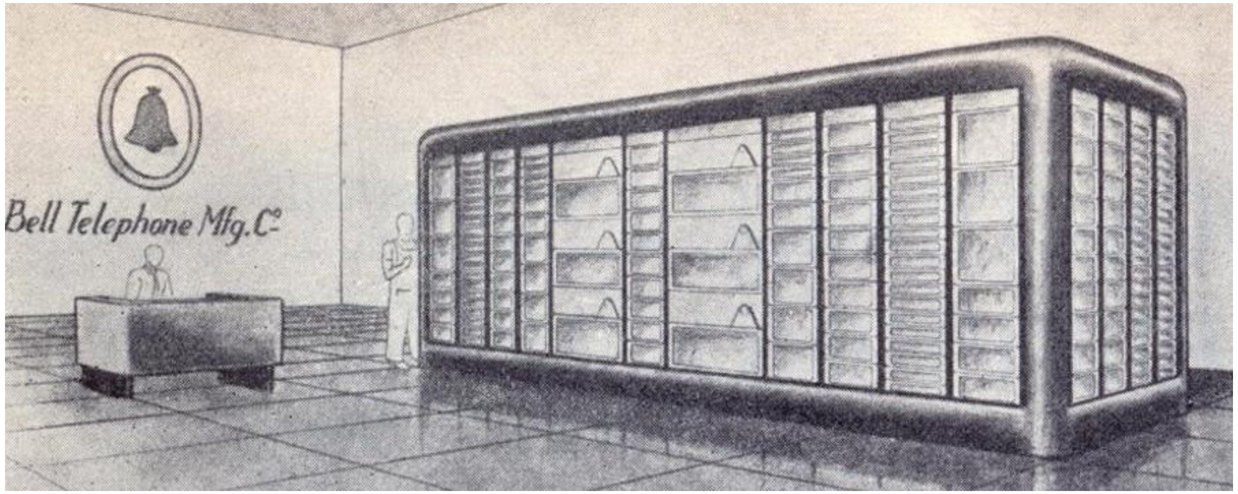
FOSDEM



Early Electronic Computing in Belgium: Analysis and Simulation of the IRSIA FNRS Mathematical Machine

Christophe Ponsard, Marie Dudekem Gevers
NAM-IP Computer Museum

FOSDEM 26 – Retrocomputing DevRoom – February 1

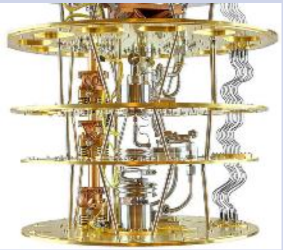
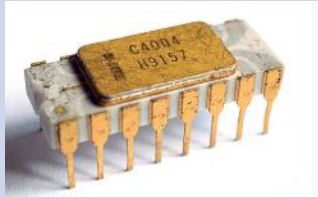
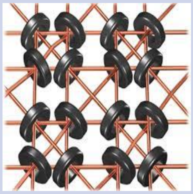
NAM-IP Computer Museum

Computer Museum NAM-IP

www.nam-ip.be

- Located in Namur/Belgium - 30’ from Brussels
  - worth a visit if you are staying a few days I Belgium after FOSDEM)
  - also: HomeComputerMuseum (in Eindhoven/NL not so far), Mémoire Morte (Lille)
- Missions:
  - Preservation: safeguarding digital heritage, focus on local pioneers
  - Acquisition of artefacts, enriching collections: Bull, Burroughs/Unysis, I&B,…
  - Exhibitions: for all, specific animation, permanent/temporary
  - **Research: about machines, software, communities especially local (Belgium)**
- “Container design”, an historical parallel

Back in time to 1st generation computer (before Moore's "law")

Generation	5th →	4th →	3rd →	2nd →	1st
When	Now→future?	1971 - now	1964-1971	1956-1963	1940-1956
Technology	AI ? Quantum ?	**Microprocessor (1971)** (V)LSI, DRAM, SRAM (optical disks)	**Integrated circuit (1958)** Better cores, HDD, tapes	**Transistor (1947)** **Mag. core memory** **HDD, mag. tapes**	Vacuum tube Magnetic drums Delay lines, tapes, cards
Computers		Microcomputers: IBM, Apple,… SUN stations,…	IBM 360 (mainframe) PDP, VAX (mini)	IBM 1401 Bull Gamma 60	ENIAC, EDVAC IBM 701, Bull Gamma3 MMIF
#gates	More Moore ?	10^4-10^{10}	10^3-10^5	10^3-10^4	10^3-10^4
	 	 	 	  	  

“Computer” circa 1950

- Human (NACA 1949) [notice gender]

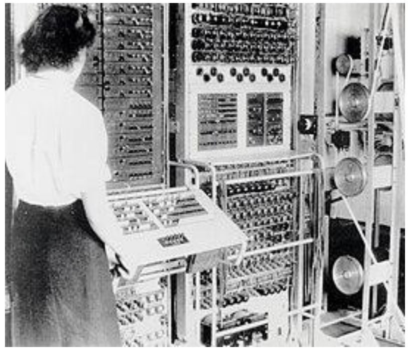
- (Electro)mechanical calculator/computers

IBM 407

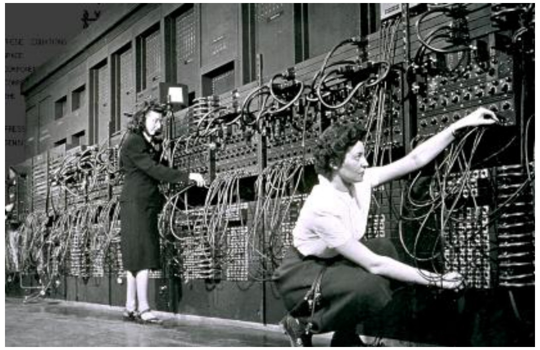
Bull BS120

Harvard Mark I (with Aiken)

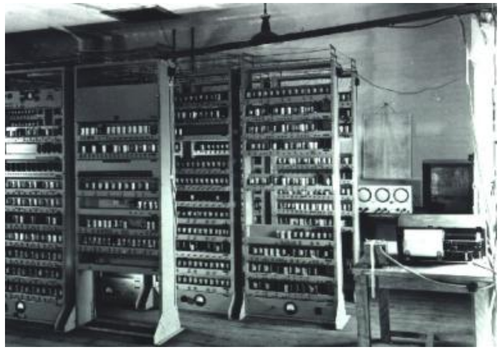
- Early electronics computer (wired or stored program)



Colossus (1943-45)



ENIAC (1945→ 1948)



EDSAC (1949)

Mathematical Machine ?

- **Purpose: scientific and technical computation rather than business data processing**

  = solve numerical problems efficiently
  - by carrying out a series of arithmetic operations, sometimes using tables of functions.
  - E.g. system of linear equations, high degree, equation, differential equations
- Can be called mathematical machines:
  - Colossus (wired): cryptanalysis (breaking German Lorenz code during WWII)
  - Harvard Mark (wired): math. tables, scientific calculations, ballistics, and rocket research
  - Bell Labs Model III (wired): ballistic computing
  - ENIAC (wired/stored): artillery tables, scientific computing, weather,…
  - EDSAC (stored): heat, planet orbits, ballistics, math tables,…
  - **MMIF (Belgian),** BARK (Sweden, electromechanical)
- **Requirements (digital approach >< analogic)**
  - High precision arithmetic operations (fixed or floating point)
  - Math library: set of core functions to compute trigonometric functions, exp/log, extract roots,…
  - Algorithmics: control flow
  - Matrix: index support

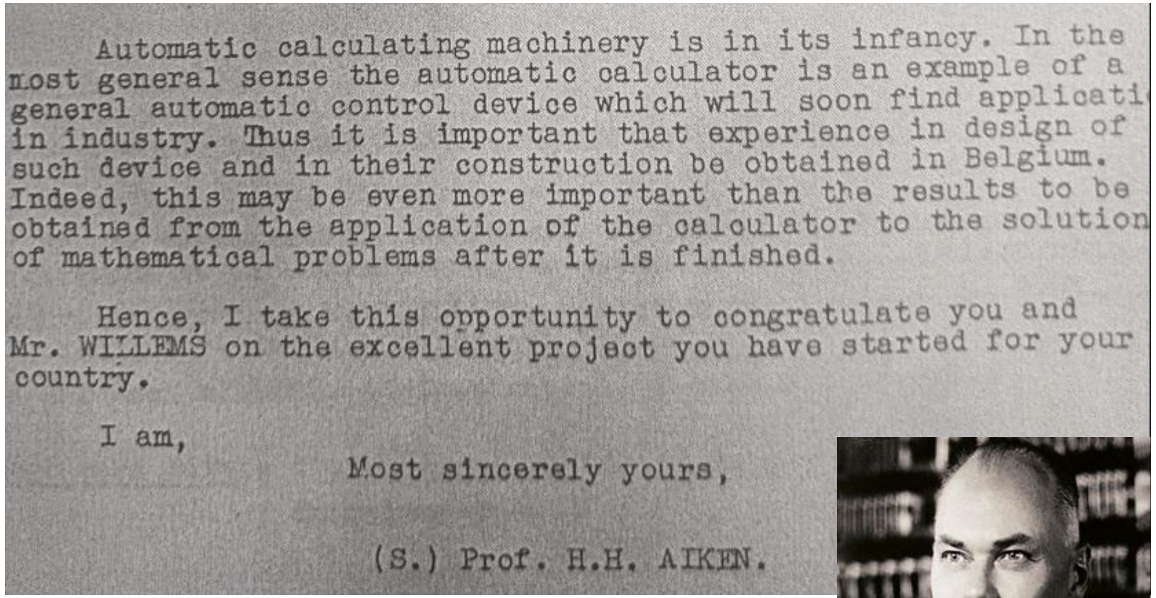
Belgian Context and Approach for MMIF

Post-WWII Context:

- **National Technological Autonomy: develop own expertise** in electronic computing rather than relying solely on foreign machines.
- **Scientific Advancement:** support research in mathematics, physics, and engineering, providing a powerful tool for numerical computations.
- **Capacity Building and Training:** train Belgian scientists, engineers, and mathematicians in the design, construction, and use of electronic computers.

Approach:

- **Survey (1946–1947):** research fund FNRS sent scientists to the USA to study “large mathematical machines”
- **Hands-on Experience (1947–1948):** Belgian engineers and scientists assisted in building the Harvard Mark III making also connection with Howard Aiken
- **Support and governance:** IRSIA/FNRS funding CECE organization, Aiken consultancy



Automatic calculating machinery is in its infancy. In the most general sense the automatic calculator is an example of a general automatic control device which will soon find applicati in industry. Thus it is important that experience in design of such device and in their construction be obtained in Belgium. Indeed, this may be even more important than the results to be obtained from the application of the calculator to the solution of mathematical problems after it is finished.

Hence, I take this opportunity to congratulate you and Mr. WILLEMS on the excellent project you have started for your country.

I am,

Most sincerely yours,

(S.) Prof. H.H. AIKEN.

Timeline

- 1936 – Turing Machine concept: 1936 (Turing)
- 1944 – Harvard Mark 1: 1944 (Haiken)
- 1945 – Von Neumann architecture, ENIAC “V2”
- 1947 – Transistor (Bell Labs), Assembly code (ARC, K. Booth)
- 1949 – EDSAC (first stored program computer)
- 1951 – Ferranti Mark I (first commercial, manual by Turing!)
- 1952 – Bull Gamma 3, IBM 701
- 1953 – Magnetic core (2^{nd} generation)
- 1957 – IBM 704 (scientific with floating point)
- 1959 – IBM 1401 (2^{nd} generation)
- 1964 – IBM System/360 (3^{rd} generation)

MMIF ERA

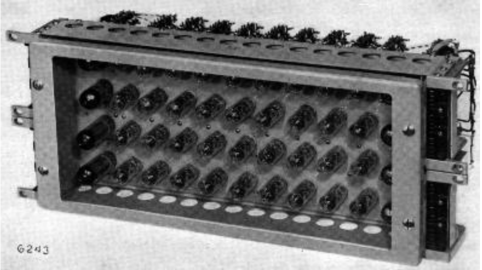
1946-1951	Ideas
1952-1955	Design Prototype
1956-1957	Full version
1957-1962	Exploitation

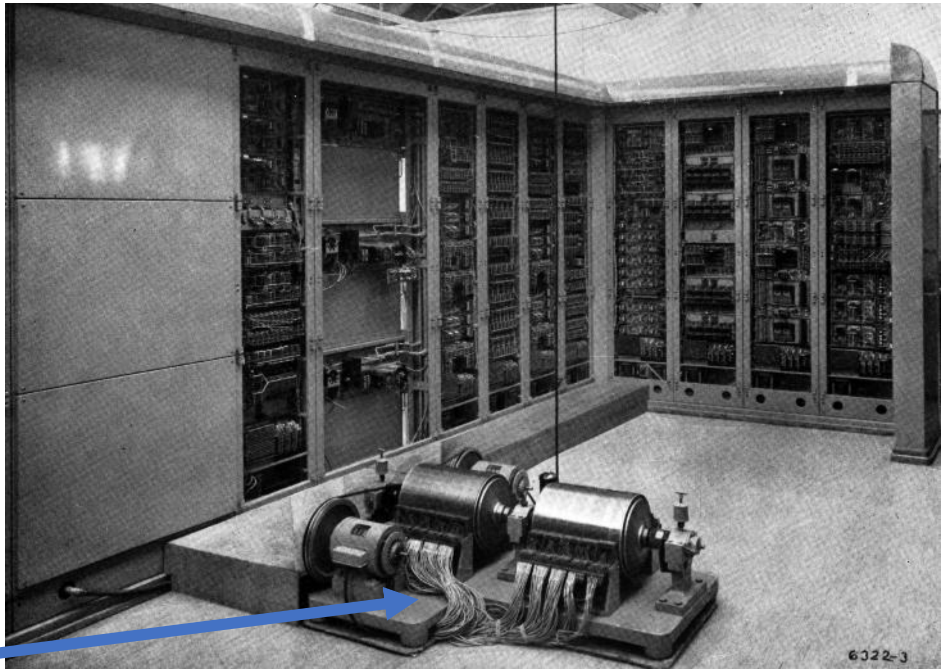
Physical View

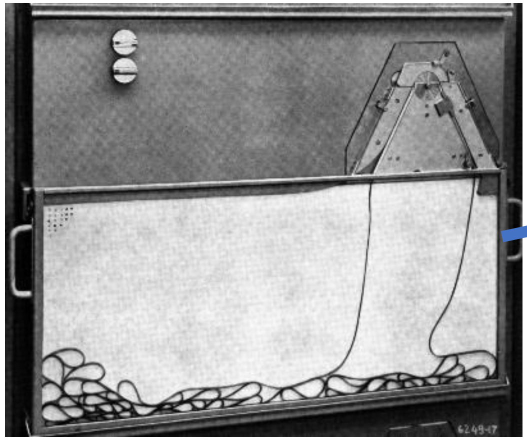
- Physically: L shape
  - Initial (17 racks): 7.50 m X 2.50 m X 2.50 m
  - Final (34 racks): 13 m

- Hardware:
  - 3000→5000 hot vacuum tubes
  - **Very fast electronic memory “register”**
    delay line based on gaz tubes
    18 (decimal) digits length
  - **Fast memory (7ms)** = 2 drums
    1 data + 1 program (Harvard style)
    100 tracks x 20 sectors x 18 digits
    (~ 19 KB each)
  - **Slow memory**: 6 « infinite » tapes
  - 1 console
  - 1 printer

- Consumption
  - 15 → 25 kW
    = small residential area in the 1950’s
    so very hot→ cooling







Logical Architecture : “Harvard” architecture (1971)

Data and Program separation

Also used in microcontrollers
(e.g. TI handheld games)

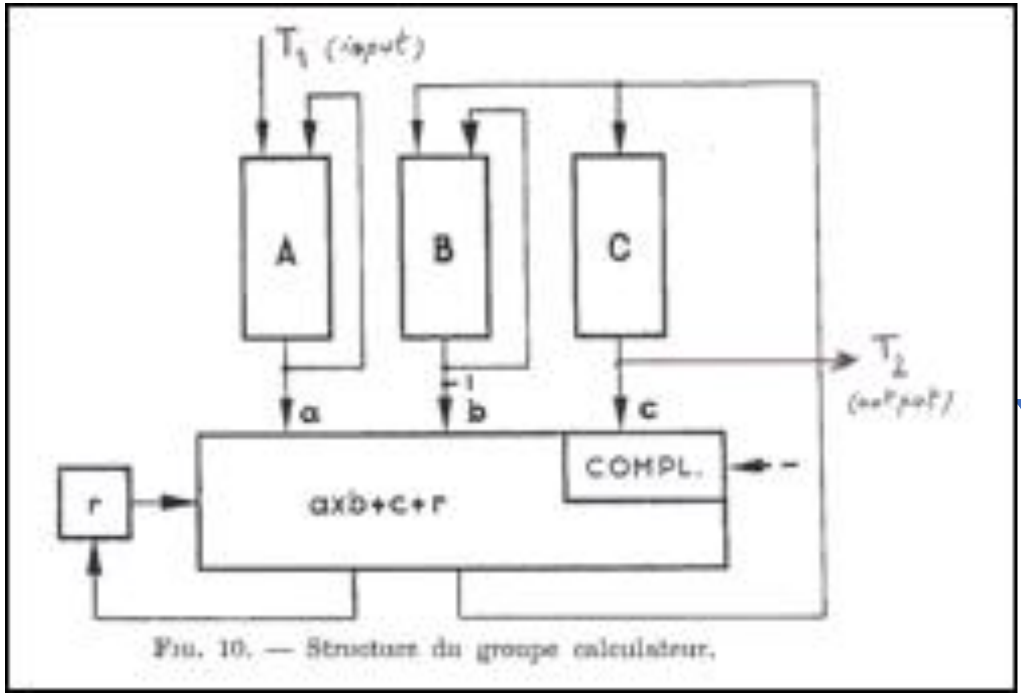
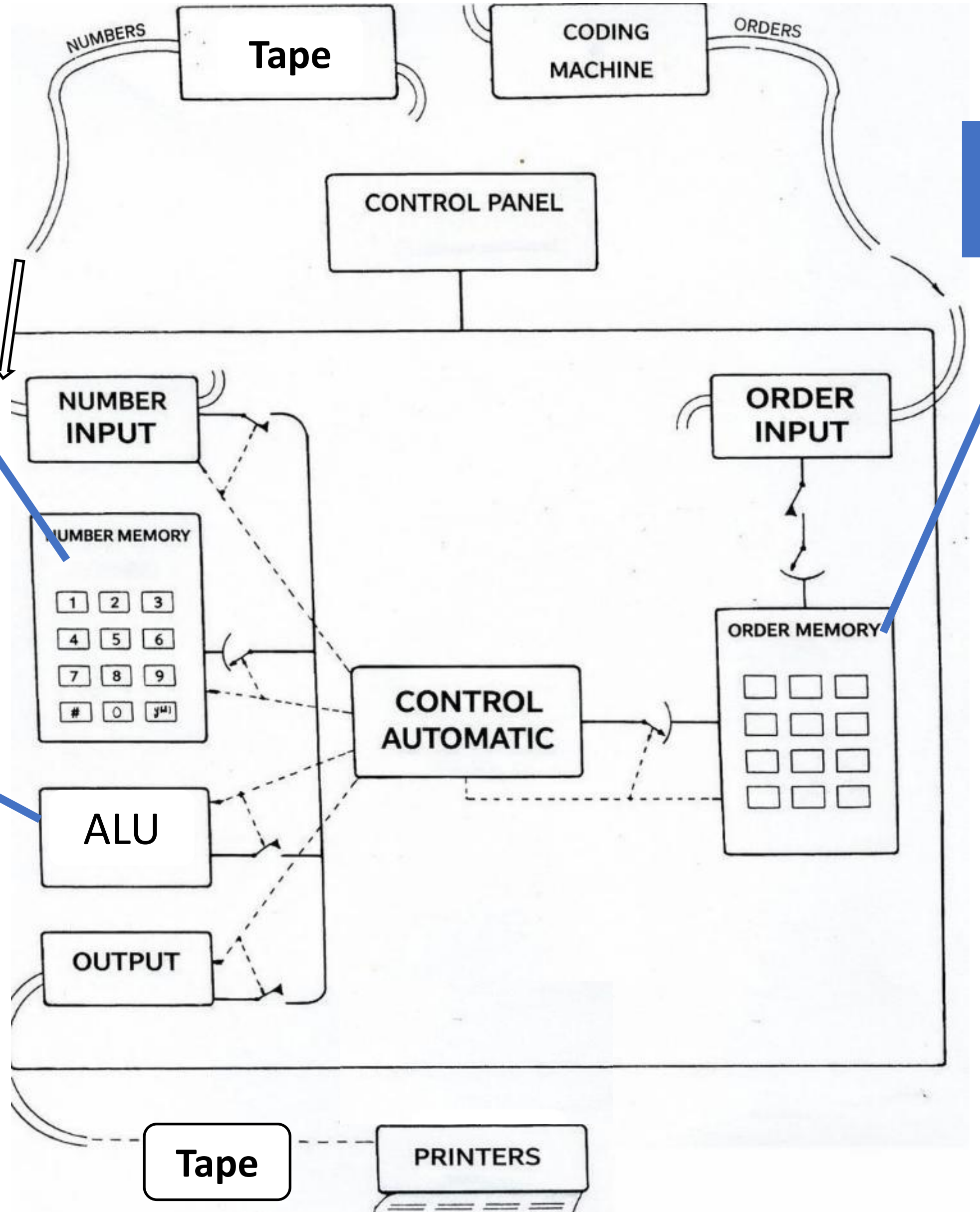


Fig. 10. — Structure du groupe calculateur.

ALU: a x b + c in one cycle (~ drum access time, 7-14ms)
- using “9” complement, x using tables
<u>No division !</u>



[No clock]

Data and Instruction Representation

- Word = 18 digits (called tetrads because 4 bits)
  - using decimal representation, not binary
    - debated that "binary" is better ➔ need to adopt some time
      decimal is better we avoid conversion, etc
    - Physical coding: biquinary : like in Colossus
      less power and more resilient
      (not too many 1, not too close)

	abcd		abcd
(0)	0000	(5)	1000
(1)	0001	(6)	1001
(2)	0010	(7)	1010
(3)	0101	(8)	1101
(4)	0100	(9)	1100

- Represents either a number or a pair of instructions
  - Number representation:
    - Float by default: (sign) (15 digits mantissa) (sign) (2 digits) exponent (~IEEE precision)

$$\pm 0.abcd\ldots \; 10^{\pm pq}$$

/!\ two represen

    - Fixed point also supported, possible "double precision" (not investigated)

  - Pair of instructions: 1 instruction = 9 digits ➔ only impair instruction "jumpable"
    TTTTT AAAA TTTTT AAAA

Registers and Drums

Note: Greek letter translated to ASCII low case equivalent
a= α b=β w= ω

- Registers
  - floating registers - 18 digits
    - w (ω) : accumulator : directly used by the ALU as input/output
    - E, F : internal registers with fast access time
    - ~~b (β) : internal of register of calculating unit, can buffer information~~
  - index register - 4 digits
    - G, H, I, J aka Wi (i=1..4) with "i" bit used in indexed instruction scheme
  - other registers
    - ch: sign (boolean type), used for conditional jumps
    - M: ~program counter, points to the next address (+5 modulo 10000)
    - ~~V, S: 4 digits, used for target address for tape operation (to be checked)~~
    - ~~A, B: not registers but current tape value on tape A or B~~
- Drums:
  - 100 tracks numbered from 0 to 99
  - 20 sectors numbered from 0 to 95 by step of 5
  - addressing α = TTSS: 4 digit (= second part of instruction)
    - 1215 means track 12, sector 15
    - 1211 is illegal

Instruction set (minimal core)

- **Global format : type (5 digits) + address/immediate (4 digits)**
  - **Type part**

Instruction type/subtype (transfer, computation,		Index in many operations	Instruction dependent	

  - **Number part:** most of the time address: data (read/write) or instruction (jump)
    for some instructions: immediate value (constant)

- **ALU operations**

Example:
04016 9000
ω ← ω+data[9000]

Mnemo	0	Data source	Aux. register	Operation type	Normalisation
<op>F <op>E <op>(v) <op>(v+Wi) <op>[a] <op>[a+Wi]		2: from register 3: Immediate address → .abcd as number index register is added 4: from memory (+ possible index)	F: 0 E: 1 0 no index 1..4: index 0 no index 1..4: index	1: + 2: - 3: x 4: x- 6: + 7: -* 8: * (using β) 9: *- (using β)	5: not normalised (+,-) 6: normalised

Instruction set (minimal core)

- **Write accumulator (ω) back in drum memory/E/F at given address α (a)**

Mnemo	0	Target	Register	Operation type	6
=a		9: memory	0	4: resets ω	
->a				6: keeps ω	
->E		7: register	0:F		
=E			1:E		
->F					
=F					
=ch			4:ch (sign)		

- **Jumps**

Mnemo	4	Type	0	0	0
rv		0: unconditional		4: keep ω	
rv-		1: if ch negative		6: reset ω	
end		2: end program			

Subtlety: comparison patterns

-a+b >< +b-a 😅

Rv- si a ⩾ b	Rv- si a > b	Rv- si a ≤ b pour b ≠ 0	Rv- si a < b
- a	+ b	- b	+ a
+ b	- a	+ a	- b
= ch	= ch	= ch	= ch
Rv-	Rv-	Rv-	Rv-

Case a<=b where b can be 0 is left to reader 😅

Subtlety: « alteration » instruction 00xxx

- **instruction that will alter NEXT instruction !**
- alteration is applied **ONCE** on ω prior to performing the operation
- so virtual: takes no time
- Examples:
  - Extract mantissa
  - Change sign
  - Module (abs value)
  - Discard mantissa
  - Read/write exponent

préfixe	symbole	effet
00012	+ man	+ abcd... ; + 00
00062	- man	- abcd... ; + 00
00022	+ sgn	+ s 1000... ; + 01
00072	- sgn	- s 1000... ; + 01
00032	+ mod	+ abcd... ; σ pq
00082	- mod	- abcd... ; σ pq
00042	+ exp	+ 1000... ; σ pq
00092	- exp	+ 1000... ; - σ pq
00013	moex	+ 1000... ; s ab
00023	exmo	σ pq00... ; + 02

Old and modern tools

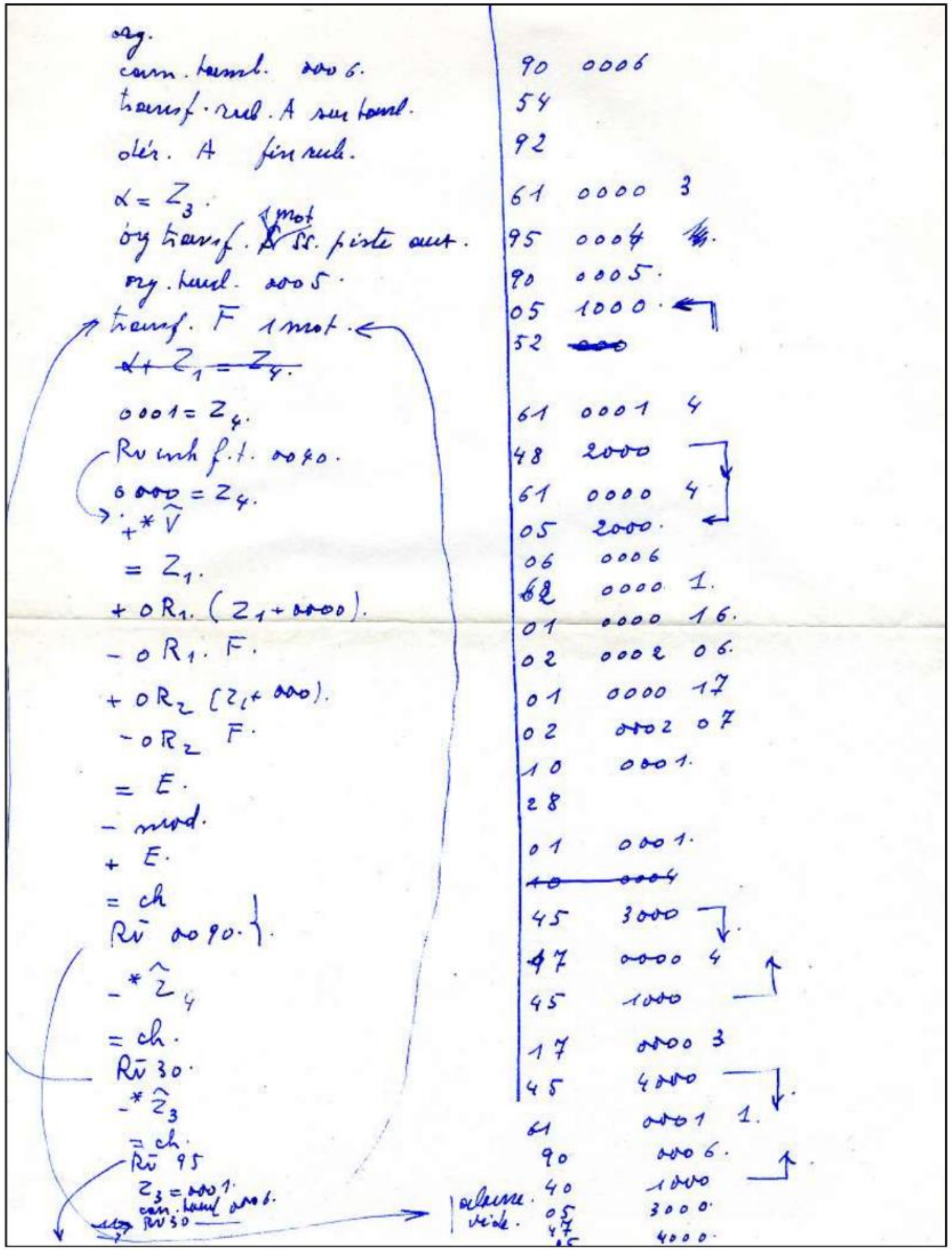
- Back in the days: no full compiler
  - Mnemonic for design
  - Translated to high level pseudo code
  - Utility for final translation

- Developed tools: avoided pseudo code
  - **Decompiler** from code to mnemonic – pattern (unfold)
  - **Compiler** from mnemonic to code
  - **Code execution** using machine model
  - note: ASCII compatible mnemonics

➔Python implementation of machine model ~1 KLOC
  - Complete: ALU, control unit, floating points, alarms
  - Partial: tape I/O
  - Not yet: fixed point/double precision

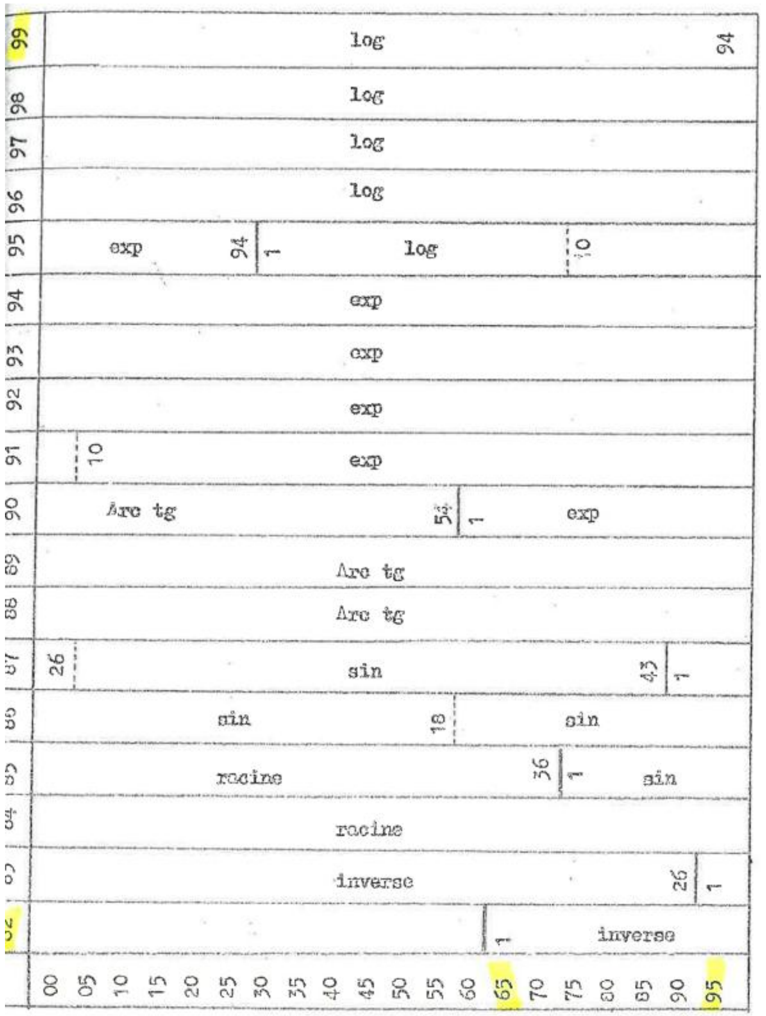
  From technical document
  (about 200 pages with some missing bits)

- **Result: raw machine with control flow and + - x ALU**
  ➔ need more math support !



Primitive Function Library

- On the drum starting address 8265
  - after 9400: required constants and specific work cells
  - can be erased/restored from a tape
    to use memory for other purposes
- Set of "primitive" functions (scientific paper)
  ➔ easily generate all others
  - 1/x, sqrt
  - sin(x) ➔ cos(x), tan(x)
  - arctan ➔ arcsin, arccos
  - exp/log



- Using iterative numerical algorithms e.g. ~Newton-Raphson
- Time to compute: 0.3s for 1/x ➔ 1.5s for exp/log)

Demo: computing 1/x ?

Documented
but let's disassemble
library code:

8265	09096 9945	07046 0000
8270	03076 0003	09046 9950
8275	00012 0000	02016 0000
8280	04036 9450	04016 9455
8285	00092 0000	02036 0000
8290	00022 0000	02036 0000
8295	07196 0000	02035 0000
8300	04016 9460	02136 0000
8305	07196 0000	02035 0000
8310	04016 9465	02136 0000
8315	07196 0000	02035 0000
8320	04016 9470	02136 0000
8325	07196 0000	02035 0000
8330	04016 9405	02136 0000
8335	07196 0000	02035 0000
8340	04016 9405	02136 0000
8345	07196 0000	04036 9945
8350	04076 9400	07046 0000
8355	04076 9440	00032 0000
8360	02066 0000	07446 0000
8365	02116 0000	41400 0000
8370	07146 0000	04016 9945
8375	07046 0000	04066 9950
8380	03066 0001	09096 9950
8385	07446 0000	41000 8275
8390	47000 0000	00000 0000

```
8265 09096 ->a      9945
8265 07046 =F
8270 03076 -*(v)    0003
8270 09046 =a       9950

8275 00012 +man
8275 02016 +F
8280 04036 x        9450
8280 04016 +        9455
8285 00092 -exp
8285 02036 xF
8290 00022 +sgn
8290 02036 xF
8295 07196 ->E

8295 02035 x-F
8300 04016 +        9460
8300 02136 xE
8305 07196 ->E

8305 02035 x-F
8310 04016 +        9465
8310 02136 xE
8315 07196 ->E

8315 02035 x-F
8320 04016 +        9470
8320 02136 xE
8325 07196 ->E

8325 02035 x-F
8330 04016 +        9405
8330 02136 xE
8335 07196 ->E

8335 02035 x-F
8340 04016 +        9405
8340 02136 xE
8345 07196 ->E

8345 04036 x        9945
8350 04076 -*       9400
8350 07046 =F
8355 04076 -*       9440
8355 00032 +mod
8360 02066 +*F
8360 07446 =ch

8365 02116 +E
8365 41400 rv-K

8370 07146 =E
8370 04016 +        9945
8375 07046 =F
8375 04066 +*       9950
8380 03066 +*(v)    0001
8380 09096 ->a      9950
8385 07446 =ch
8385 41000 rv-      8275

8390 47000 end.al
8390 00000 noop
```

3 trials

(x*1/x)-1 should be close to 0 !

OK return from function

If <3 trials then try again else alarm

8265	1	→ r_1 = F	x en case de travail x en F
8270	2	-* 0003 = r_2	nombre de reprises autorisées
8275	3	+ man + F	première approximation
8280	4	× a_0 + b_0	
8285	5	- exp × F	
8290	6	+ sgn × F	
8295	7	→ E	y_0 = (sgn x) (-exp x) (a_0 man x + b_0)

Rough initial estimation (using alterations)

5 iterations: $y_{i+1}=y_i\,(a_i-x.y_i)$ with a_i~2 (tuned)

8350	18	× x_1 -* 1 = F	contrôle F = (xy - 1)
8355	19	-* T + mod	addition et soustraction sans normalisation
8360	20	+* F = ch	ω = -T + |xy - 1| si (-), on a |xy - 1| ≤ T et contrôle correct

If check fails

Demo: let’s execute code (or see live) ➔ 1/4= ?

```
-----------------------------------------
w: 4.0 E:0.0 F:0.0
8265 09096 ->a     9945
4.0
-----------------------------------------
w: 4.0 E:0.0 F:0.0
8270 07046 =F
w: 4.0
F: 4.0
-----------------------------------------
w: 0.0 E:0.0 ,
8270 03076 -*(v)   0003
a: 0.0  b:0.0003  res:-0.0003
-----------------------------------------
w: -0.0003 E:0.0 ,
8275 09046 =a      9950
-----------------------------------------
w: 0.0 E:0.0 ,
8275 00012 +man
-----------------------------------------
w: 0.0 E:0.0 ,
8280 02016 +F
a: 0.0  b:0.4  res:0.4
-----------------------------------------
w: 0.4 E:0.0 ,
8280 04036 x       9450
a: 0.4  b:-49.689441  res:-19.8757764
-----------------------------------------
w: -19.8757764 E:0.0 ,
8285 04016 +       9455
a: -19.8757764  b:54.6583851  res:34.7826087
-----------------------------------------
w: 34.7826087 E:0.0 ,
8285 00092 -exp
-----------------------------------------
w: 34.7826087 E:0.0 ,
8290 02036 xF
a: 34.7826087  b:0.010000000000000002
res:0.347826087
-----------------------------------------
w: 0.347826087 E:0.0 ,
8290 00022 +sgn
-----------------------------------------
w: 0.347826087 E:0.0 ,
8295 02036 xF
a: 0.347826087  b:1.0  res:0.347826087
-----------------------------------------
w: 0.347826087 E:0.0 ,
8295 07196 ->E
-----------------------------------------
w: 0.347826087 E:0.347826087 ,
8300 02035 x-F
a: 0.347826087  b:4.0  res:-1.391304348
-----------------------------------------
w: -1.391304348 E:0.347826087 ,
8300 04016 +       9460
a: -1.391304348  b:2.1645703099999998
res:0.773265962
-----------------------------------------
w: 0.773265962 E:0.347826087 ,
8305 02136 xE
a: 0.773265962  b:0.347826087  res:0.268962073772751
-----------------------------------------
w: 0.268962073772751 E:0.347826087 ,
8305 07196 ->E
-----------------------------------------
w: 0.268962073772751 E:0.268962073772751 ,
8310 02035 x-F
a: 0.268962073772751  b:4.0  res:-1.075848295091
-----------------------------------------
w: -1.075848295091 E:0.268962073772751 ,
8310 04016 +       9465
a: -1.075848295091  b:2.016019  res:0.940170704909
-----------------------------------------
w: 0.940170704909 E:0.268962073772751 ,
8315 02136 xE
a: 0.940170704909  b:0.268962073772751
res:0.252870262492714
-----------------------------------------
w: 0.252870262492714 E:0.268962073772751 ,
8315 07196 ->E
-----------------------------------------
w: 0.252870262492714 E:0.252870262492714 ,
8320 02035 x-F
a: 0.252870262492714  b:4.0  res:-1.01148104997086
-----------------------------------------
w: -1.01148104997086 E:0.252870262492714 ,
8320 04016 +       9470
a: -1.01148104997086  b:2.00013038
res:0.98864933002914
-----------------------------------------
w: 0.98864933002914 E:0.252870262492714 ,
8325 02136 xE
a: 0.98864933002914  b:0.252870262492714
res:0.250000015597714
-----------------------------------------
w: 0.250000015597714 E:0.252870262492714 ,
8325 07196 ->E
-----------------------------------------
w: 0.250000015597714 E:0.250000015597714 ,
8330 02035 x-F
a: 0.250000015597714  b:4.0  res:-1.00000006239086
-----------------------------------------
w: -1.00000006239086 E:0.250000015597714 ,
8330 04016 +       9405
a: -1.00000006239086  b:2.0  res:0.99999993760914
-----------------------------------------
w: 0.99999993760914 E:0.250000015597714 ,
8335 02136 xE
a: 0.99999993760914  b:0.250000015597714
res:0.249999999999998
-----------------------------------------
w: 0.249999999999998 E:0.250000015597714 ,
8335 07196 ->E
-----------------------------------------
w: 0.249999999999998 E:0.249999999999998 ,
8340 02035 x-F
a: 0.249999999999998  b:4.0  res:-0.999999999999992
-----------------------------------------
w: -0.999999999999992 E:0.249999999999998 ,
8340 04016 +       9405
a: -0.999999999999992  b:2.0  res:1.00000000000001
-----------------------------------------
w: 1.00000000000001 E:0.249999999999998 ,
8345 02136 xE
a: 1.00000000000001  b:0.249999999999998  res:0.25
-----------------------------------------
w: 0.25 E:0.249999999999998 ,
8345 07196 ->E
-----------------------------------------
w: 0.25 E:0.25 ,
8350 04036 x       9945
a: 0.25  b:4.0  res:1.0
-----------------------------------------
w: 1.0 E:0.25 ,
8350 04076 -*      9400
a: 1.0  b:1.0  res:0.0
-----------------------------------------
w: 0.0 E:0.25 ,
8355 07046 =F
w: 0.0
F: 0.0
-----------------------------------------
w: 0.0 E:0.25 F:0.0
8355 04076 -*      9440
a: 0.0  b:5e-14  res:-5e-14
-----------------------------------------
w: -5e-14 E:0.25 F:0.0
8360 00032 +mod
-----------------------------------------
w: -5e-14 E:0.25 F:0.0
8360 02066 +*F
a: -5e-14  b:0.0  res:-5e-14
-----------------------------------------
w: -5e-14 E:0.25 F:0.0
8365 07446 =ch
-----------------------------------------
w: 0.0 E:0.25 F:0.0
8365 02116 +E
a: 0.0  b:0.25  res:0.25
-----------------------------------------
w: 0.25 E:0.25 F:0.0
8370 41400 rv-K
```

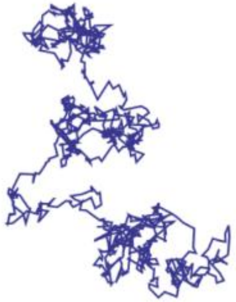
Summary of Applications Developed on the MMIF

LISTE DES PROBLEMES TRAITES PAR LE C.E.C.E.

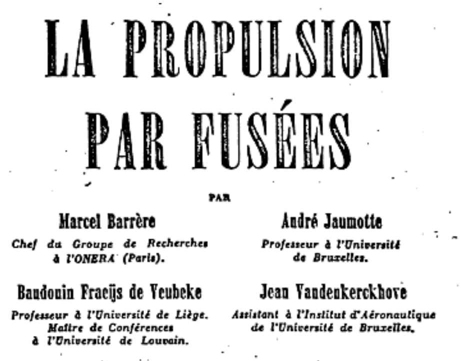
M 1	Rapports faits à Anvers jusque 1955 (direction M. Linsman)	
M 2	Fonctions de Bessel	pr l'Ecole Royale Militaire
M 3	Balistique intérieure	pr l'Ecole Royale Militaire
M 4	Trajectoire de fusée	pr le Prof. B. Fraeys de Veubeke
M 5	Erf complexe	pr le Centre de Contrôle des Rad: communications des Services mobi: (M. Marique)
M 6	Publications	tirés à part d'articles publiés par les membres du C.E.C.E.
M 7	Projets de quadratures	pr le Prof. F. van den Dungen (U.L.B.)
M 8	Problèmes anciens	posés en 1955, et non résolus sur la machine provisoire
M 9	Météorologie	I.R.M., Uccle
M 23	Statistiques philologiques	R. P. Somers (Lv)
M 23bis	Statistiques philologiques	Textes de H. Somers s.j.
M 24	Filtres passe-bande	Bell Telephone
M 25	Tables de coordonnées Distances & Azimuths	Inst. Géographique Militaire
M 26	Mouvement brownien	Prof. I. Prigogine (U.L.B.)

<u>83 problems officially tracked</u>

- Mathematics:
  - Bessel functions, matrix inversion, elliptic functions…
- Physics (research):
  - **Brownian mvt by I. Prigogine (Nobel Prize)**
  - 3 body problem
  - Astrophysics: star oscillations

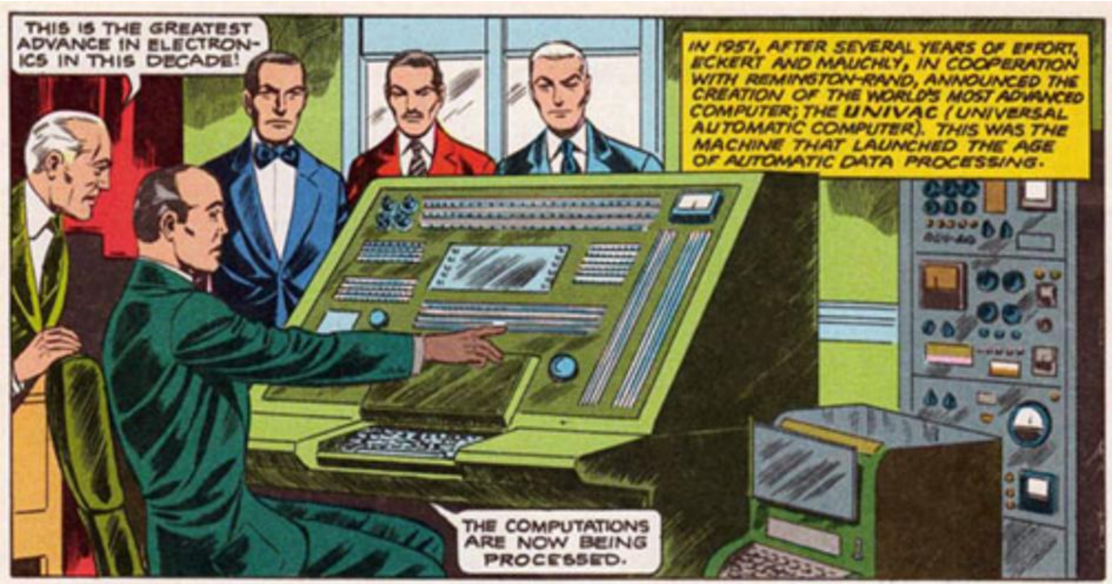


- Applied physics:
  - **Ballistics (military), Rocket trajectory (Frayes de Veubeke)**
  - Meteorology (royal institute)
  - Crystallography
  - Telecom filters
- Linguistics (early)



Conclusion

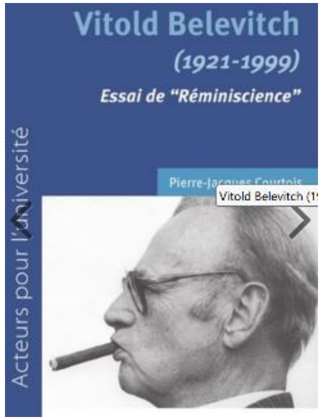
- Some familiarity:
  - Sequence of instructions
  - Instruction types, memory modes
  - Stored program (even if Harvard architecture)
- Many emerging concepts not yet fully understood/standardised
  - Weird assembly language
  - Terminology ALU, register terms not used yet
- Surprising:
  - Hardware: tubes, drums
  - Floating point as primary type
  - Biquinary coding
  - Alterations
  - No division, iterative algorithm
  - Reliability management



References → most now on archive.com

- 1947 - About Mathematical Machines by Léon Brillouin (in French)
- 1956 - About Core Functions by Vitold Belevitch (in French)
- 1957 - CECE document 1 - manuel de programmation (in French)
- 1958 - CECE document 2 - pseudo-code Manual (in English)
- 1959 - MKII document 4 - about elementary functions (in French)

- 2008 - MMIF slides by Sandra Mols (in English)
- 2010 - Account on the Machine by Pierre-Jacques Courtoy (in English)
- 2010 - MMIF Book by Marie Gevers (in French)
- 2014 - MMIF summary paper by Marie Gevers (in English)
- 2015 - Vitold Bevitch biography by Pierre-Jacques Courtoy (in French)
- 2016 – L'informatique belge de 1950 à 1970 by Jacques Loeckx (in French)



See:

- https://github.com/NAMIP-Computer-Museum/MMIF/tree/main/docs
- https://archive.org/details/mmif-tech-doc

Questions ?

Github: https://github.com/NAMIP-Computer-Museum/MMIF

Internet Archive: https://archive.org/details/mmif-tech-doc

Visit us: www.nam-ip.be mastodon: @namip@computermuseum.social

Contact me: christophe.ponsard@gmail.com masto

Remembering key people involved:

Claude Fosseprez, Frédéric Iselin, Jacques Loeckx,
Jean Meinguet, Paul Parré, Nicolas Rouche, Fritz Wiedmer,
Paul Dagnelie, Armand de Callatay,
Vitold Belevitch, Howard Aiken

Figure 17 : Au début de l'année 1955, dans la cour de l'École Polytechnique de l'ULB, cinq étudiants de la promotion de juillet 1955: de gauche à droite: Paul Dagnelie, (Guy Bridoux), André Fischer, Jacques Loeckx (et, accroupi, Robert Salade).