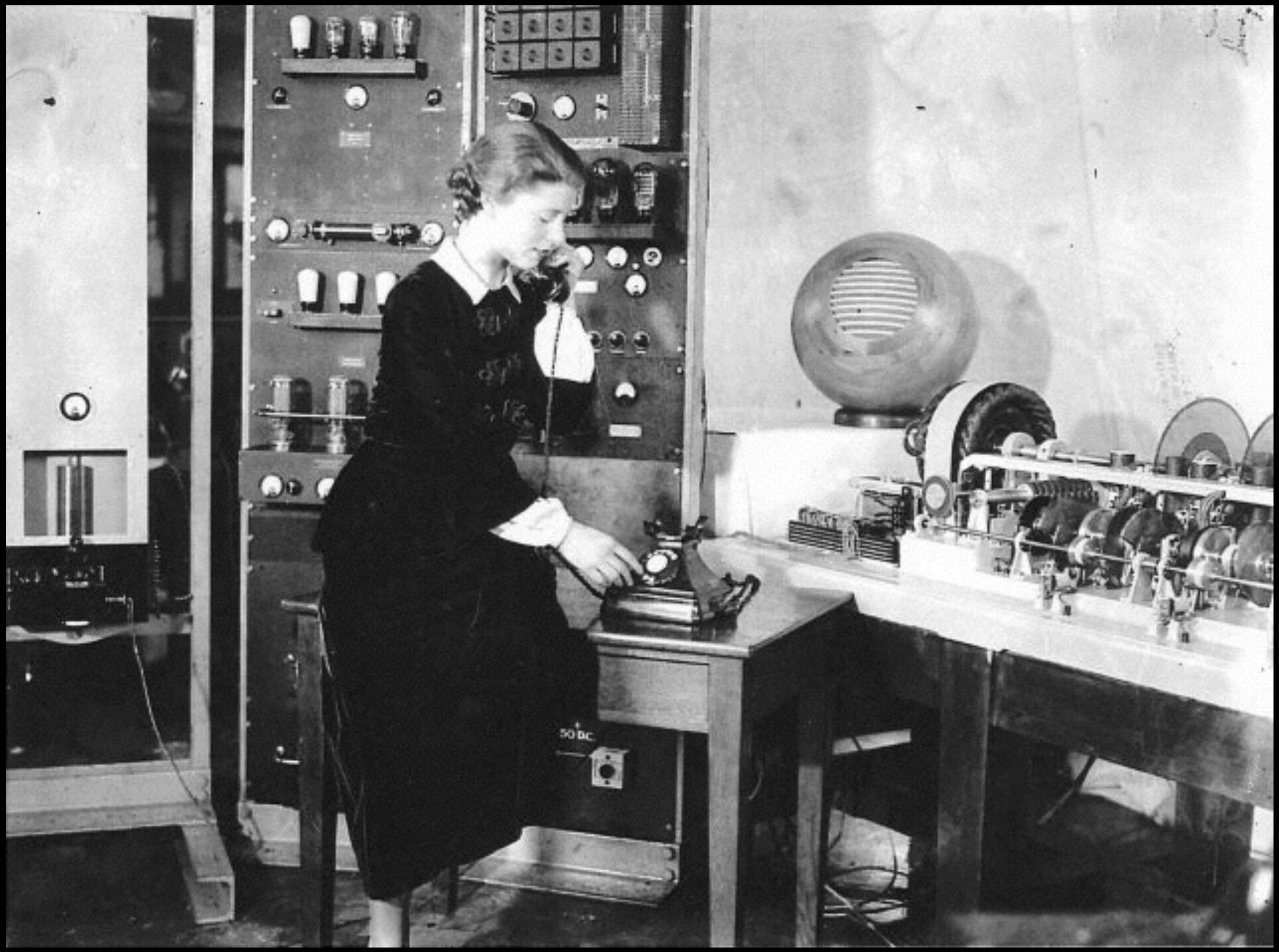
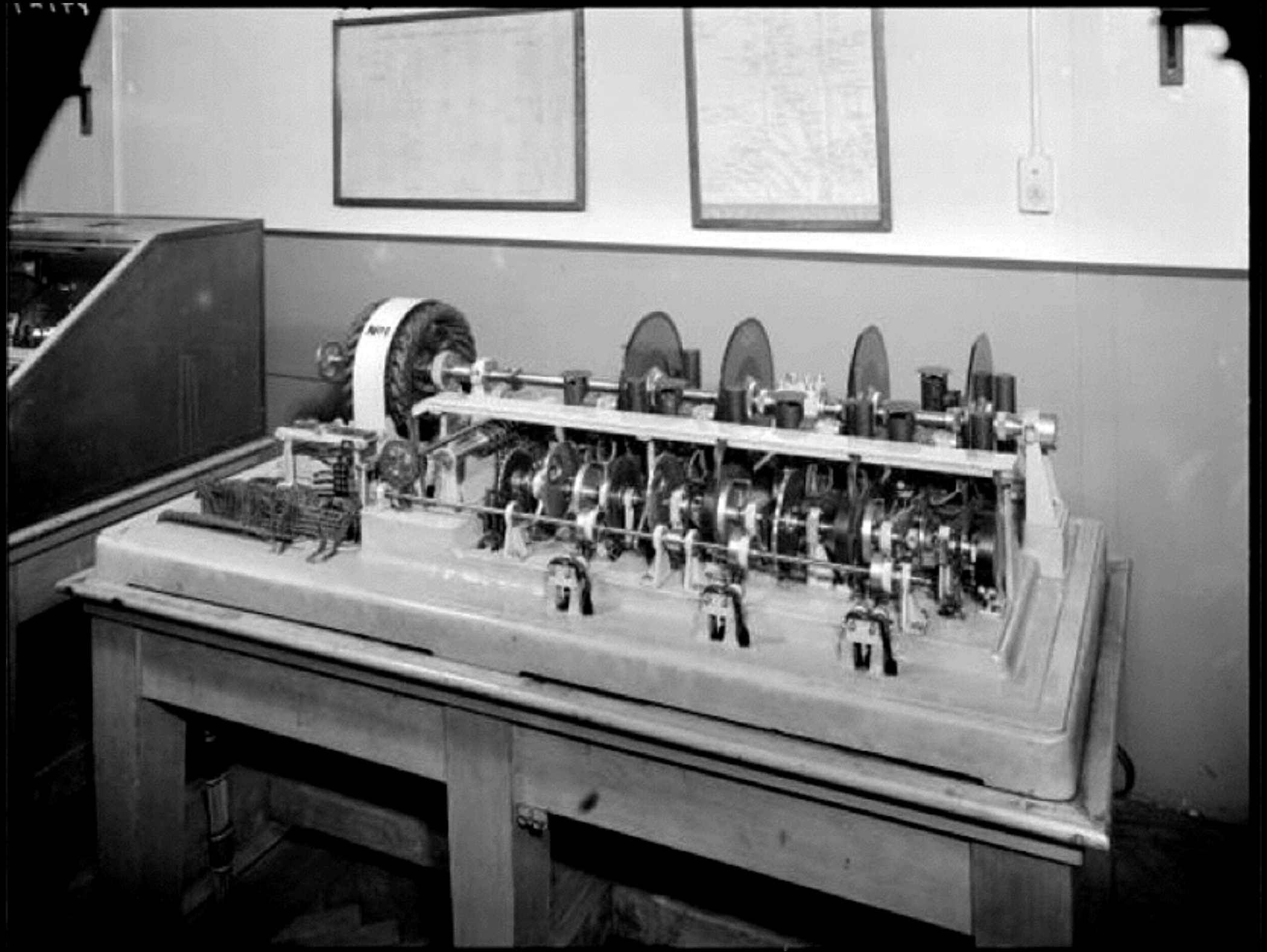
+44 20 3598 2801


50 DC



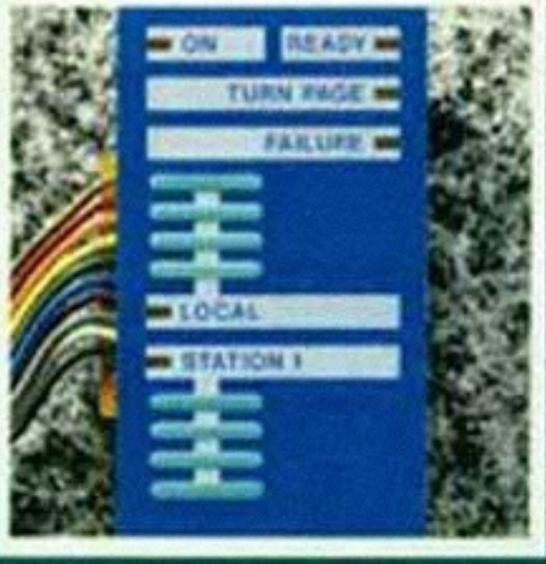

BRITISH
PATHÉ

FROM PROJECT TO PRODUCT

GIANNI BARBACETTO

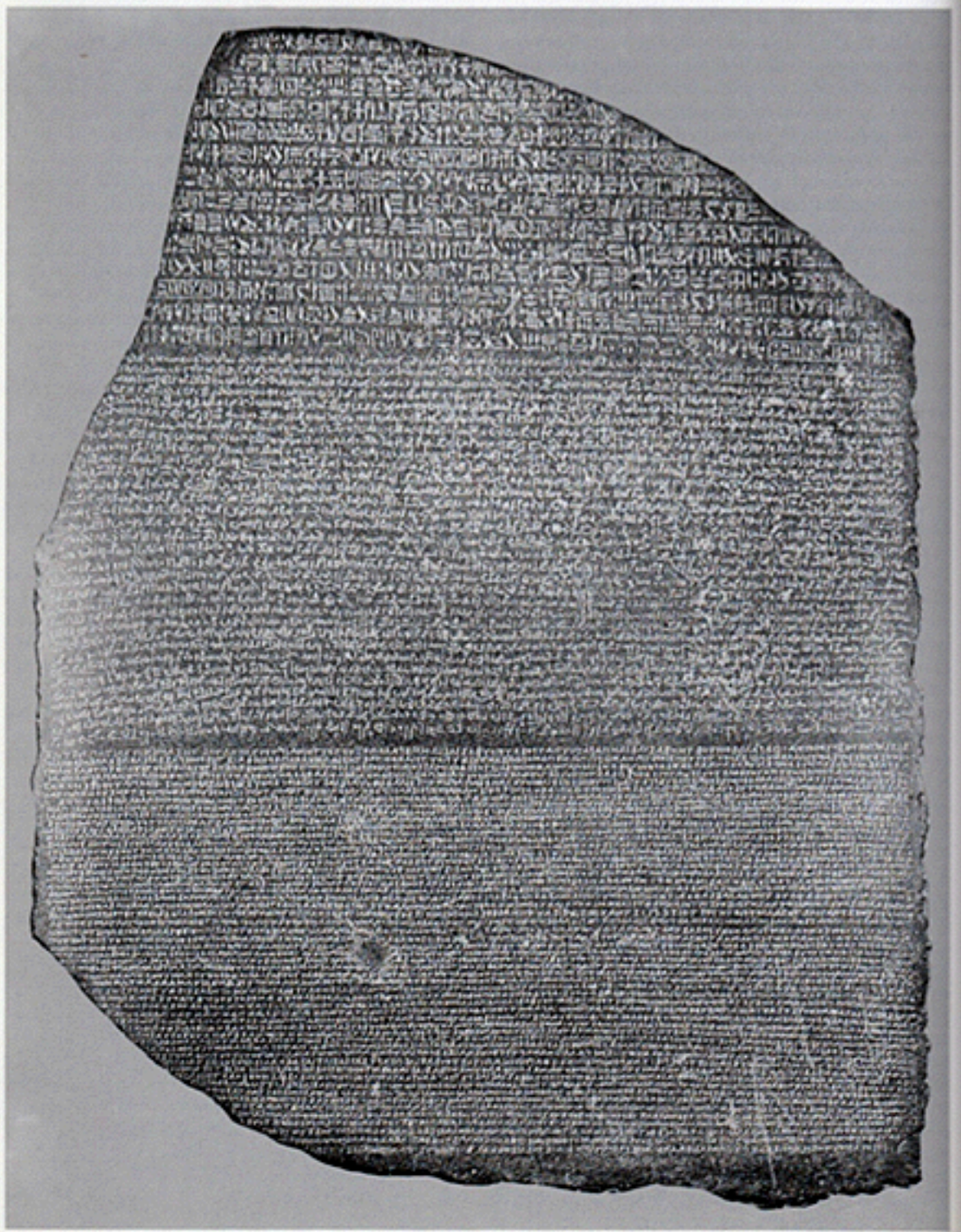
DESIGN INTERFACE

HOW MAN AND MACHINE COMMUNICATE.
OLIVETTI DESIGN RESEARCH BY KING & MIRANDA

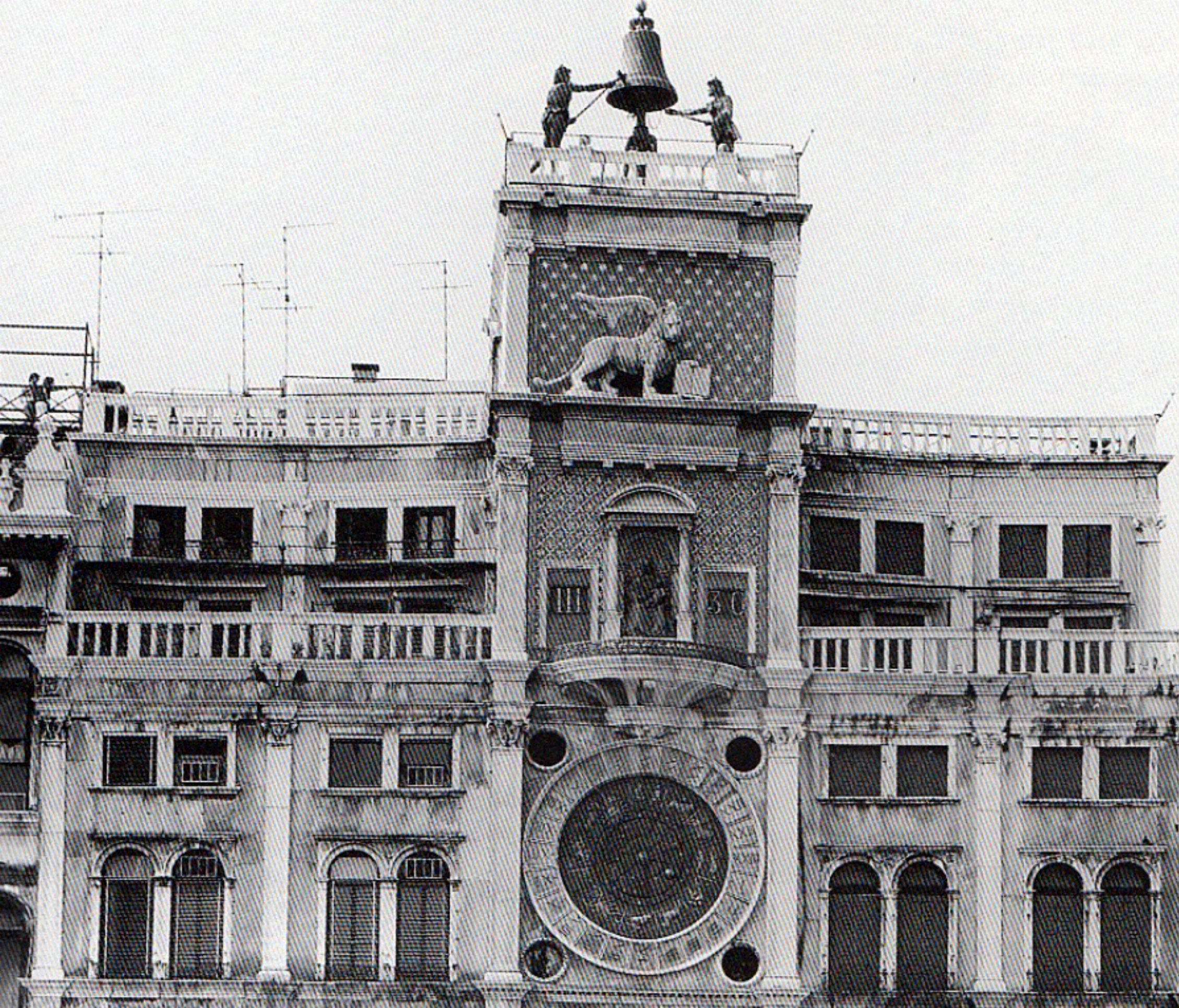


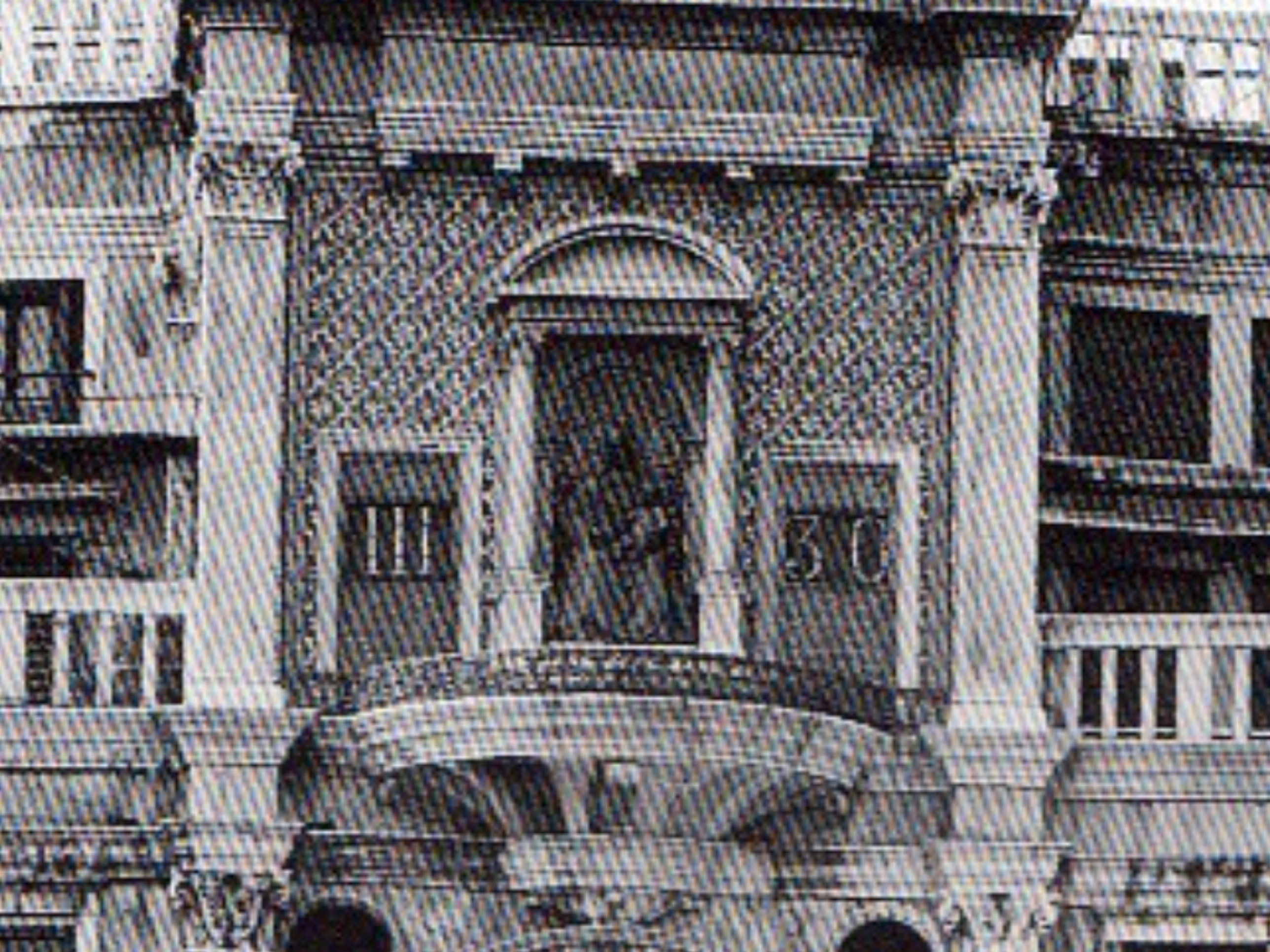
ARCADIA EDIZIONI

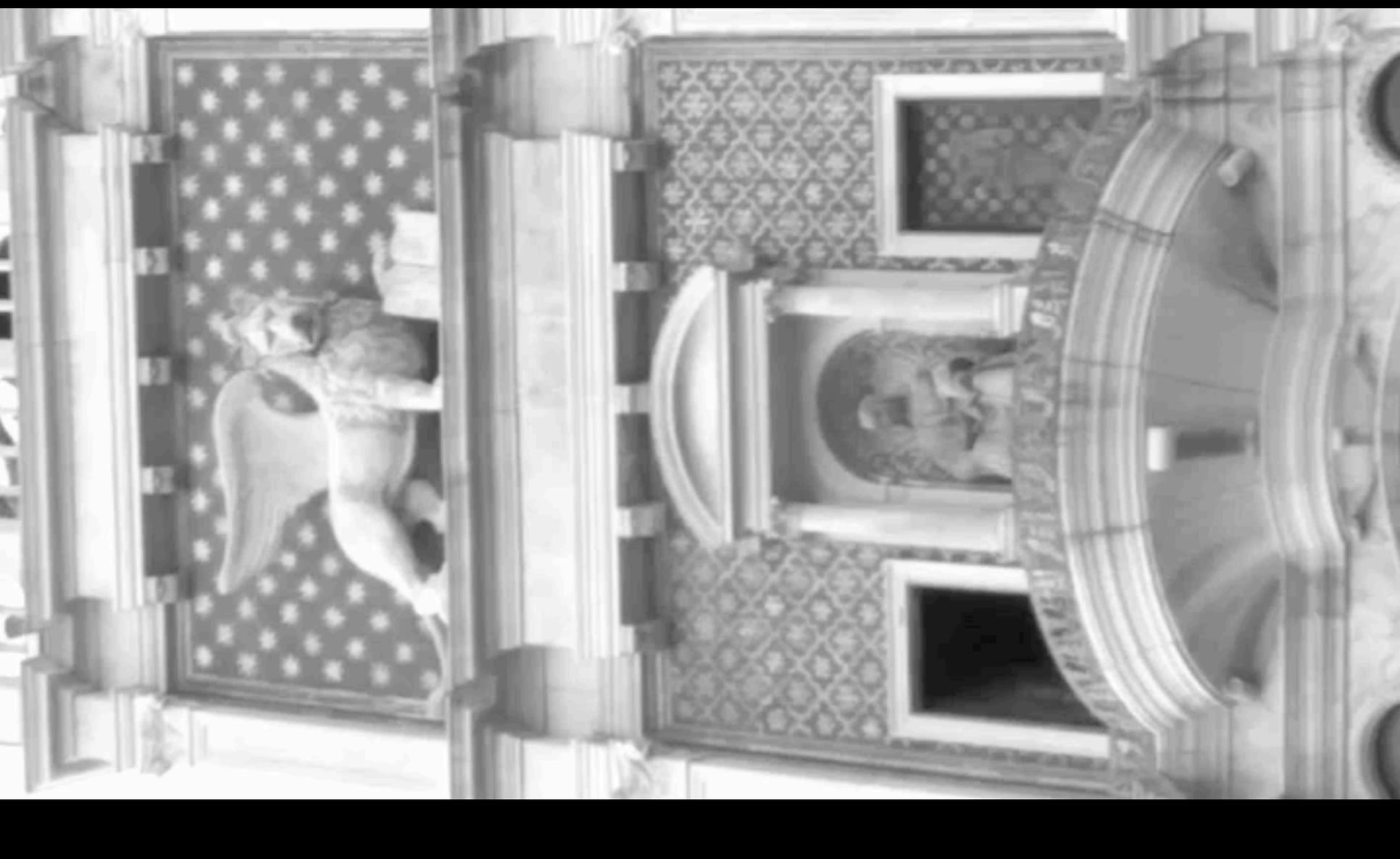
The Rosetta Stone (British Museum, London)



The Clocktower in St. Mark's Square in Venice. With public display of clocks on city towers, the first "machines" made available to the populace at large, the relationship of men with time began to change. "Cyclical," lunar, and seasonal time was replaced by an image of linear, progressive, and cumulative time.



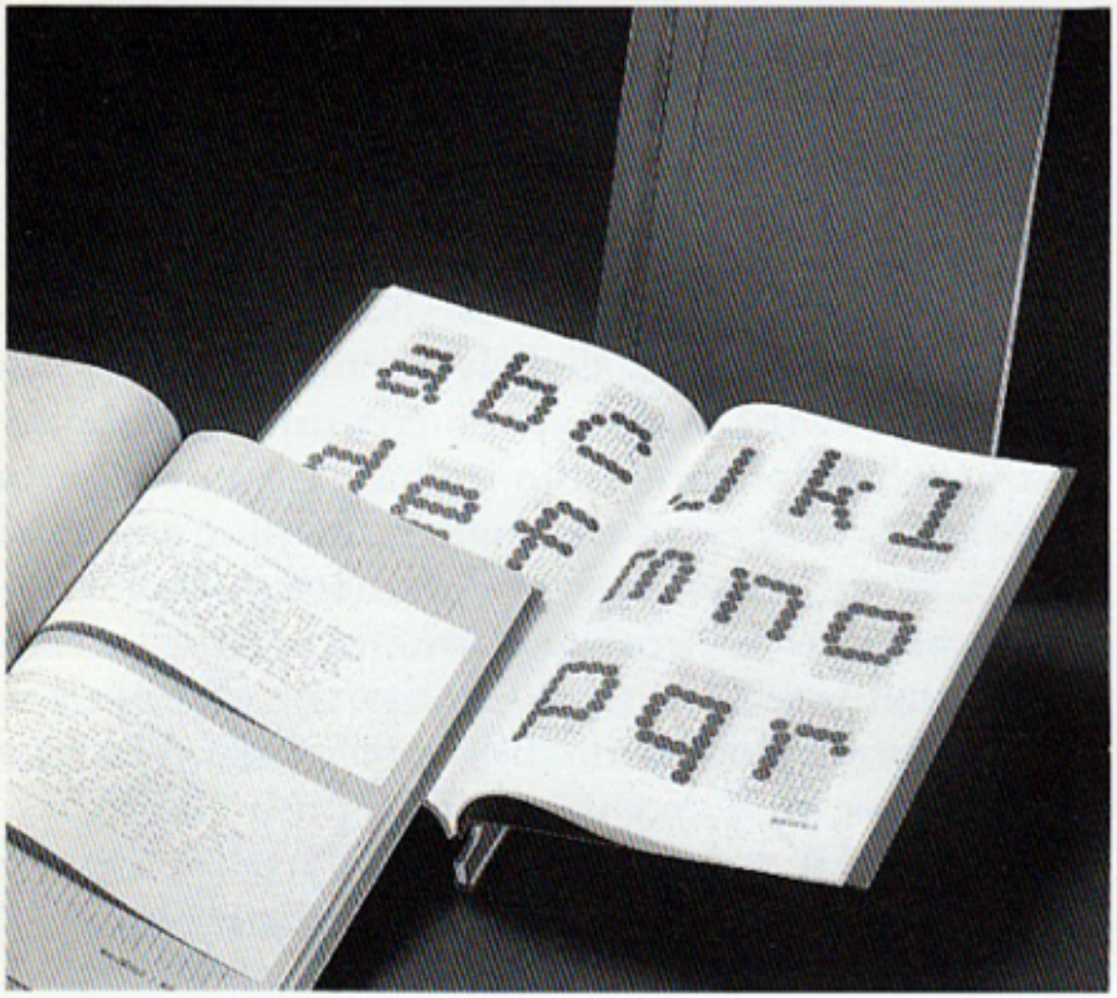




The letters of the alphabet as interface between man and machine. The ease and pleasure with which one reads improve the relationship with the user.
Left, the Olivetti dot matrix font designed by King and Miranda in 1972.
Right, several images that document the development of Olivetti writing systems (from a panel of the exhibition Design Process).

— we have not yet attained direct voice control, but a series of tools and systems (from the mouse and the joystick to the magnetic tablet and touch screen) reduce to a minimum the necessity of using codes to communicate with the machine. Now we enter more directly into contact through the use of software that is already programmed to act as a simplifying filter between the "naiveté" of the user and the complexity of the machine.
The "Macintosh case" provides a perfect example of this; the small computer produced by Apple presented itself to the user as an immediately accessible tool, with its procedures illustrated by simple drawings that appeared on the screen — while the mouse permitted the substitution of initials and coded procedures with simple hand movements that pushed the cursor across the screen; a little cartoon computer smiled when operations were performed correctly or displayed its unhappiness when an operation was botched; a small wristwatch advised the user to be patient while an operation was underway; and a trashbasket stood ready to gather data that was no longer useful, and so on. Here the design itself shifted from hard to soft, and became the design of graphic means of direct communication between the machine and the user. This is a field that is destined to grow with extreme rapidity — every interactive machine (from automatic tellers all the way to new appliances) will be able in the future to have a graphic screen for immediate dialogue with the user. The touch screen has even, in part, unified the input (tactile) and output (visual) apparatus — the screen, for certain operations indicated by video, substitutes the keyboard; the pressure of a finger on the point indicated is sufficient to obtain the operation desired.
The screen is, at any rate, along with printing terminals, an essential element of interface — between the operator and his results. Clarity of letters and words — a primary need of the user — almost always means that letters, numbers, and symbols are translated into *points* which can be easily reproduced on video screens, displays, printers; Perry King in 1972 designed the Olivetti dot-matrix font for OCR applications, later adopted by the ECMA (European Computer Manufacturers' Association). Clarity of communication also means the possiblity of communicating in one's own language with the computer, even when that means using non-Latin alphabets. Olivetti has equipped its typewriters and other systems with Cyrillic, Arabic, Greek, and Japanese (Katakana) alphabets but also with Korean, Hindi, Singhalese, Burmese, Nepalese, Thai, and Cree characters, designed at Ivrea by Arturo Rolfo.[32]
In office machines, "interface ergonomics" is imposed — more than by an intention to improve the conditions of human labor — by the necessity to organize office work and by the desire to increase productivity and efficiency. Other elements of the interface between man and machine — given the multifarious nature of the latter — cannot be grouped under these headings and are designed according to other criteria. They are: the driver's seat, an eminently public interface, which allow man to control his car while travelling, and the group of interfaces, all domestic and private, represented by the controls of household objects, commonly grouped under the class of appliances.

The letters of the alphabet as interface between man and machine. The ease and pleasure with which one reads improve the relationship with the user.
Left, the Olivetti dot matrix font designed by King and Miranda in 1972.
Right, several images that document the development of Olivetti writing systems (from a panel of the exhibition Design Process).



— we have not yet attained direct voice control, but a series of tools and systems (from the mouse and the joystick to the magnetic tablet and touch screen) reduce to a minimum the necessity of using codes to communicate with the machine. Now we finger on the point indicated is sufficient to obtain the operation desired.
The screen is, at any rate, along with printing terminals, an essential element of interface — between the operator and his results. Clarity of letters and

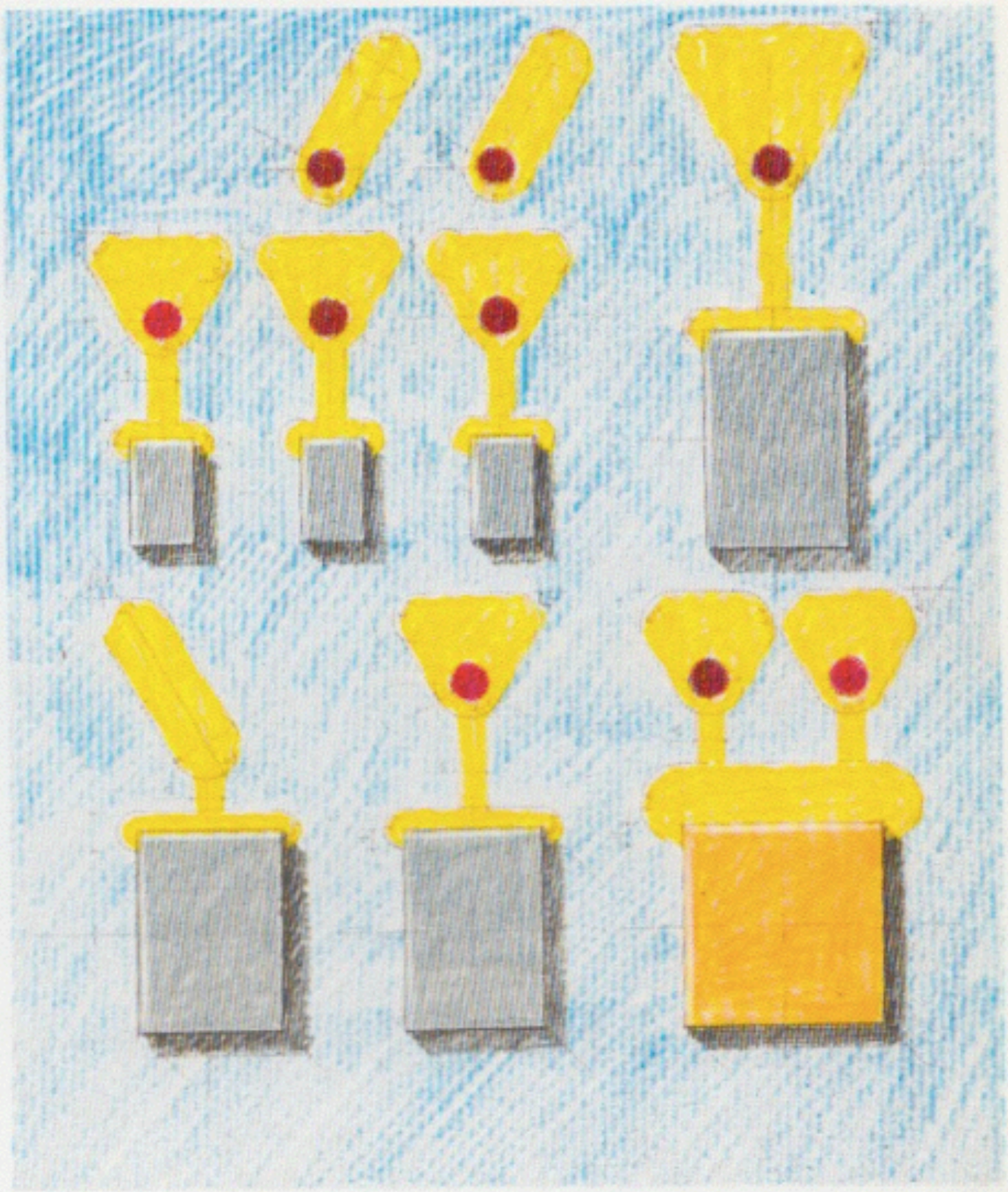
control keyboards: for
l form to the sensitive
art that allows access,
o communication with
randa prepared them-
ct for Olivetti by con-
"interface" between
itive zone in which the
the operator and his
hat Santiago Miranda
h of the electronic

l elements can be dis-

ommunicates from the
r, through a visual
ssage regarding the
given operation; this is

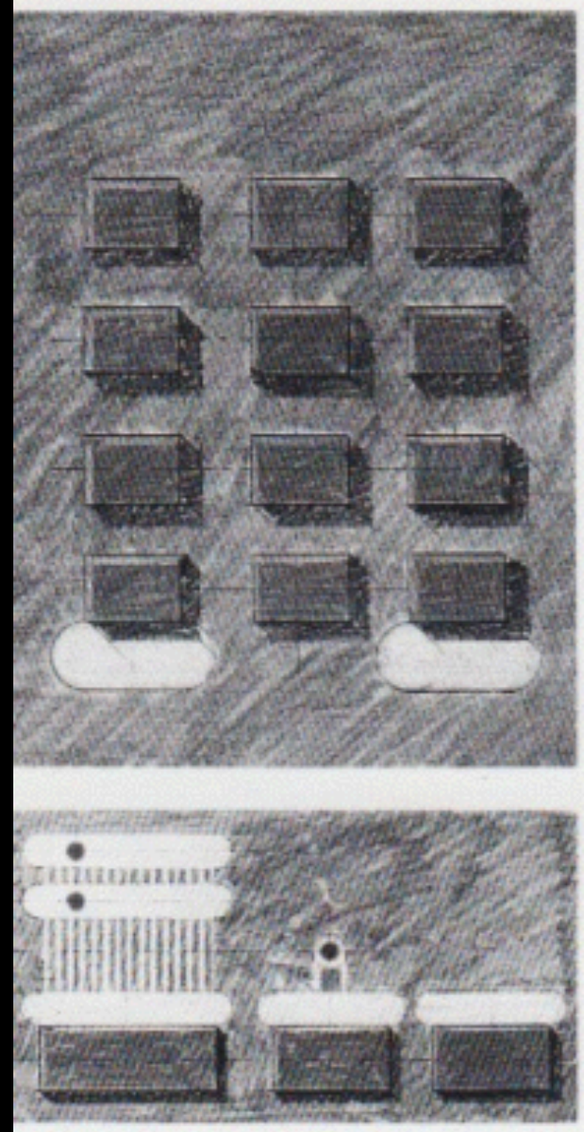
, which permits com-
oward the machine,
hich accompanies the
he target zone.

Sketch of the keyboard for the TLM 320 telecopier.

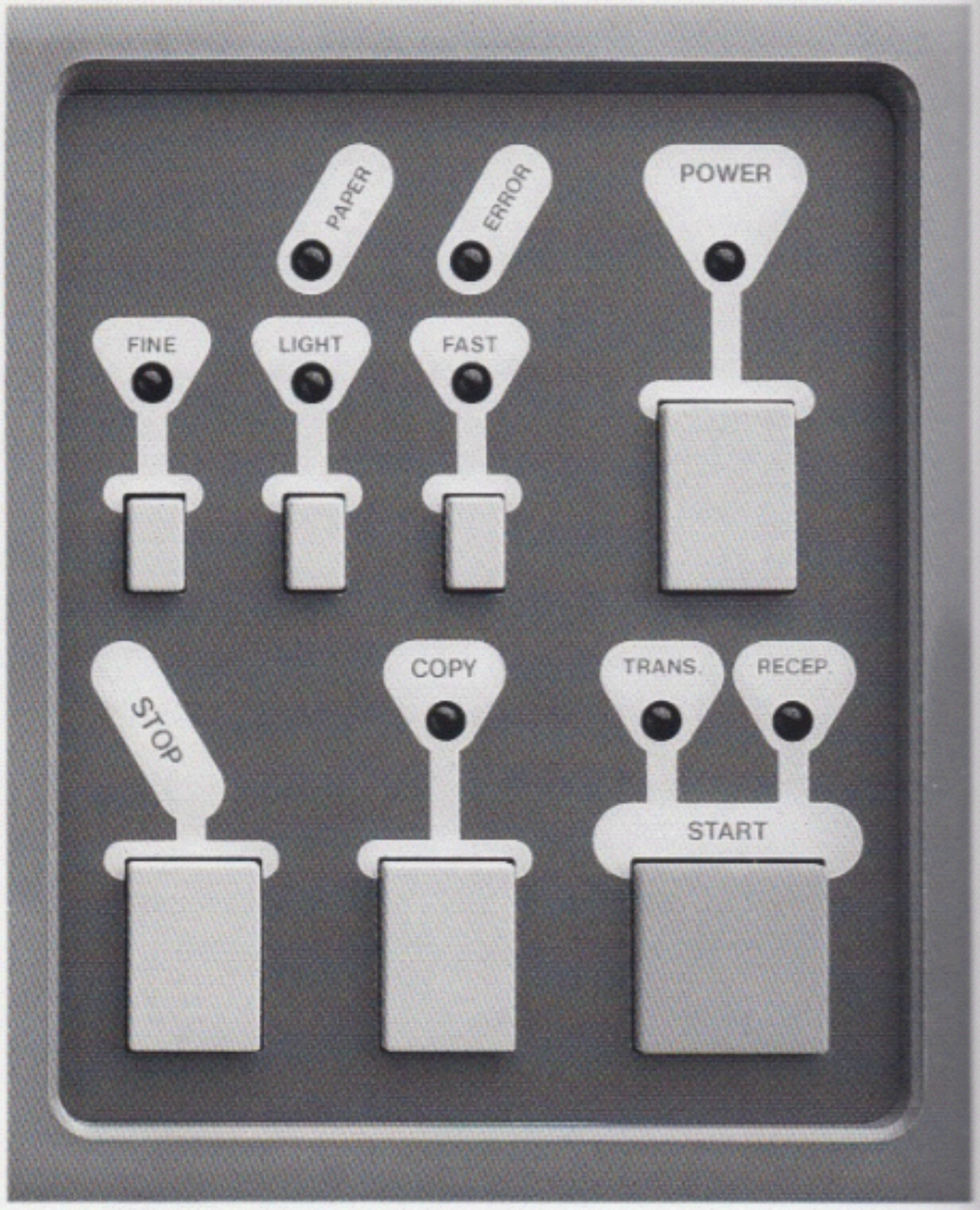


fication (legends, symbols, leds ...) of the task which

d for the TLM 332 telecopier
.



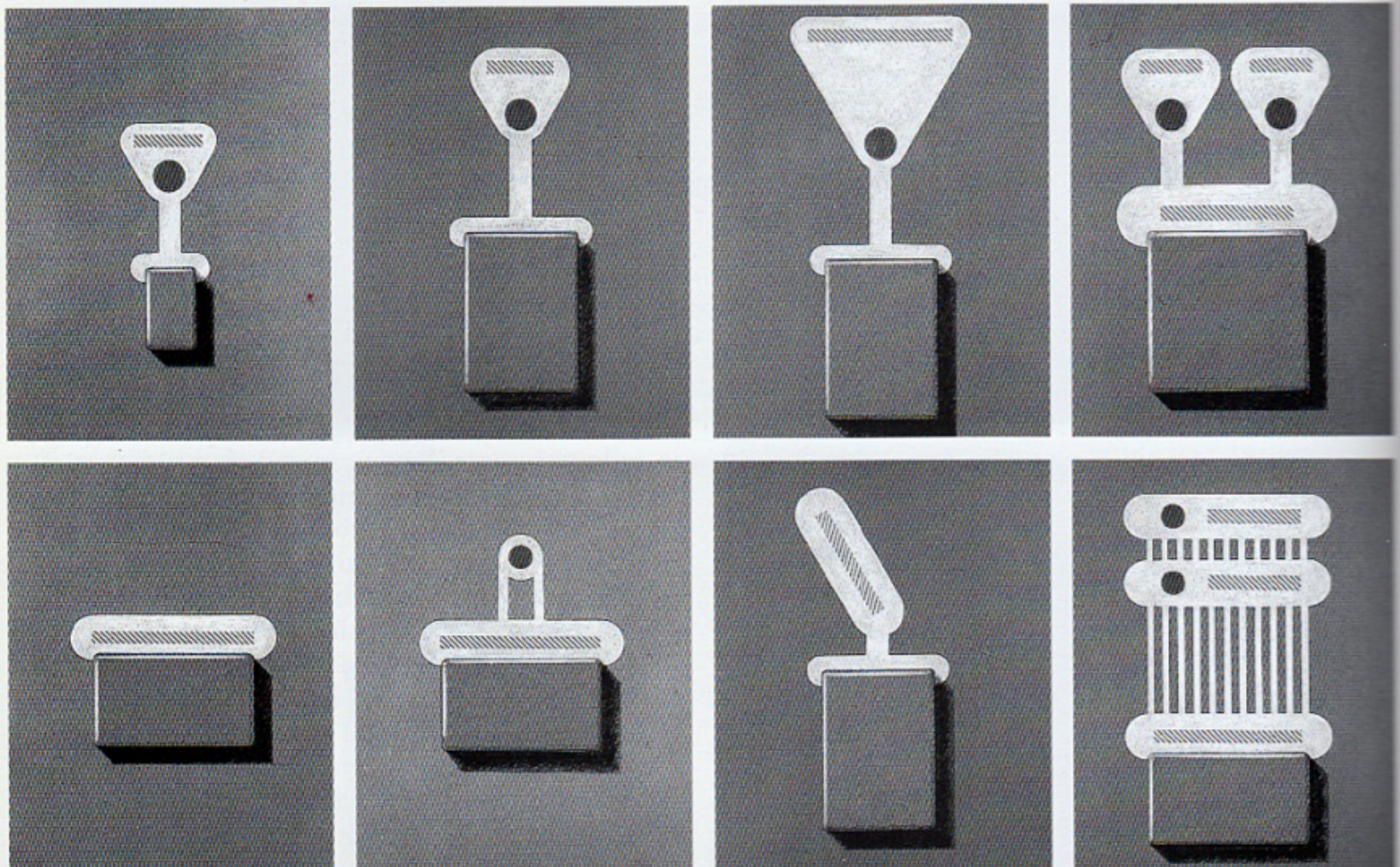
The rubber membrane keyboard for the TLM 320. The shapes, their arrangement, and the colors create the hierarchic recomposition of information.



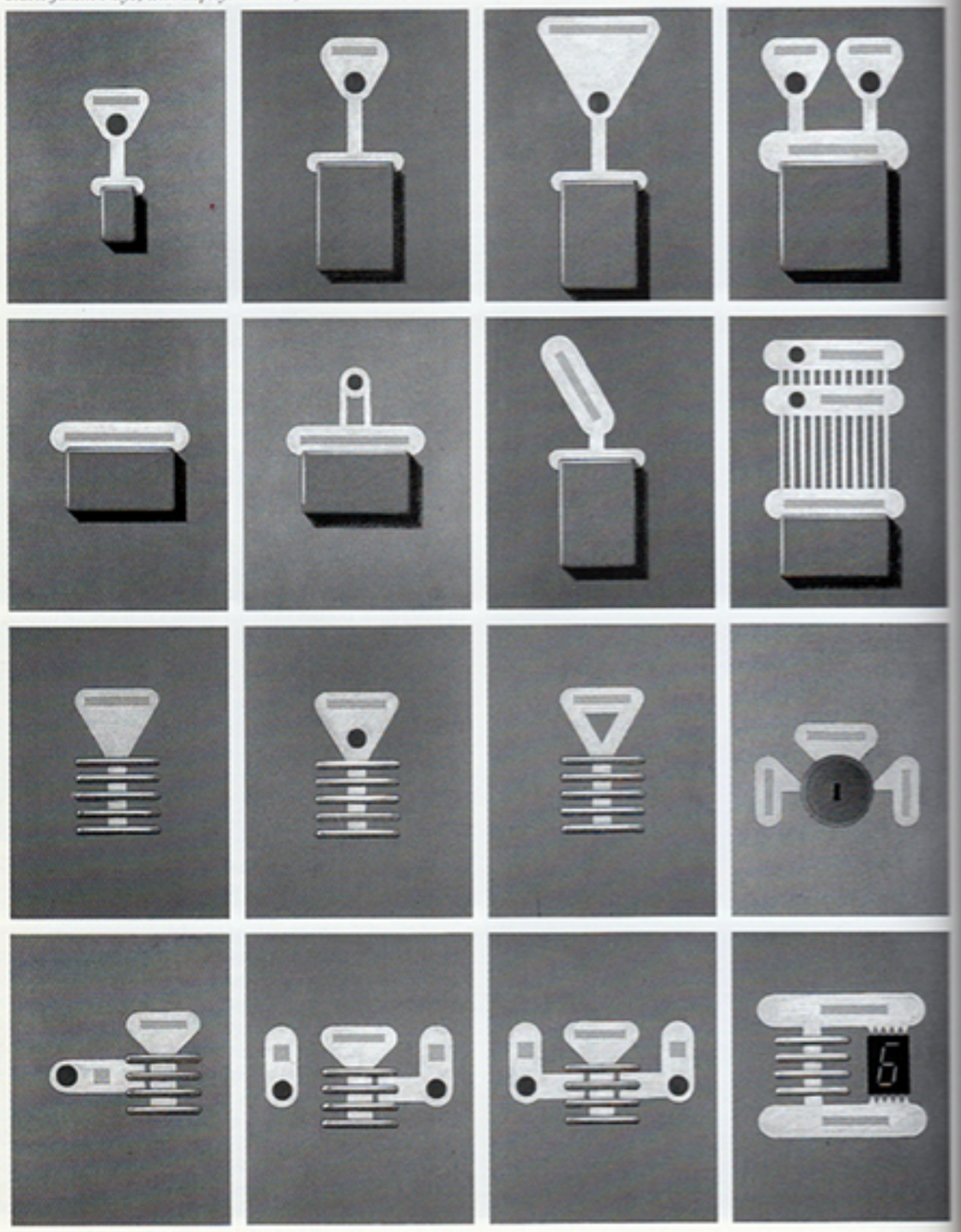
huge heavy magical
n, was controlled by a
tion of bridge-objects:
topped the gears, the
peration a mechanism
ronic circuits, the key
little hammers of a
of human fingers. The

no more than use and process, at first, elements

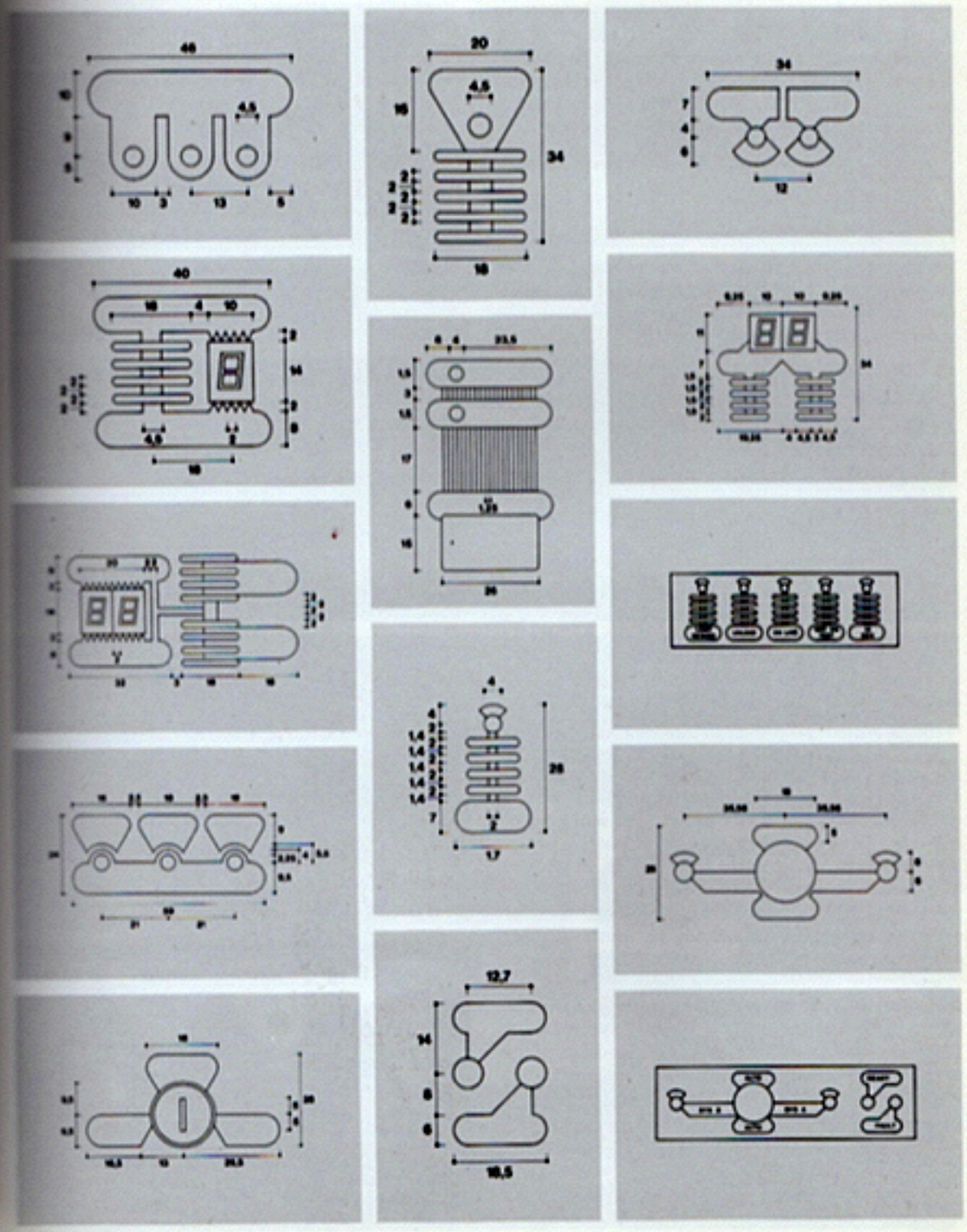
Studies for various types of three-dimensional, sensitive, simple- and double-function keys, with displays and leds, etc.



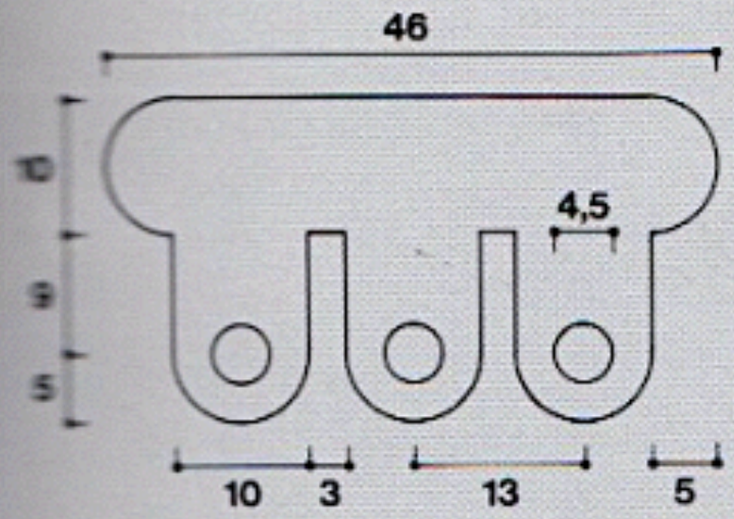
Studies for various types of three-dimensional, sensitive, simple- and double-function keys, with displays and leds, etc.

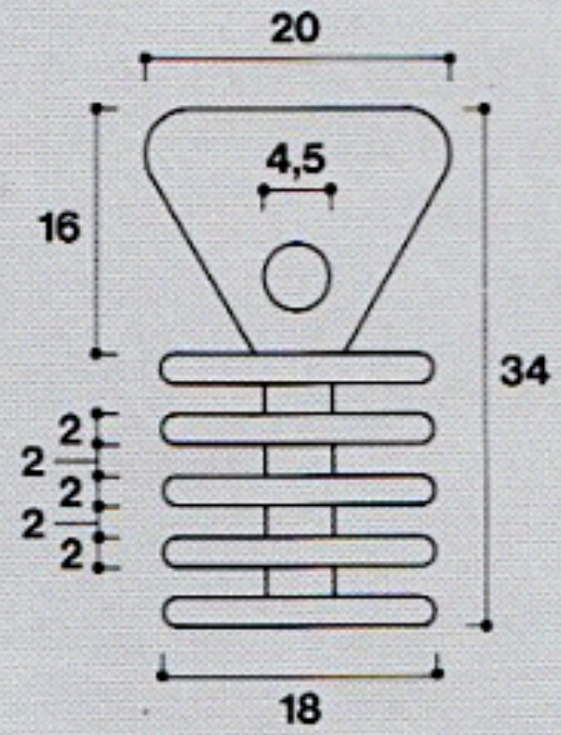


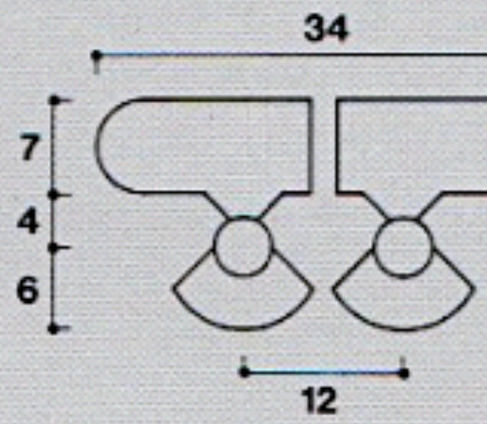
Drawings of the constituent elements of the keyboards. Just like in an alphabet of new characters, despite the smallness, each element is designed in great detail.

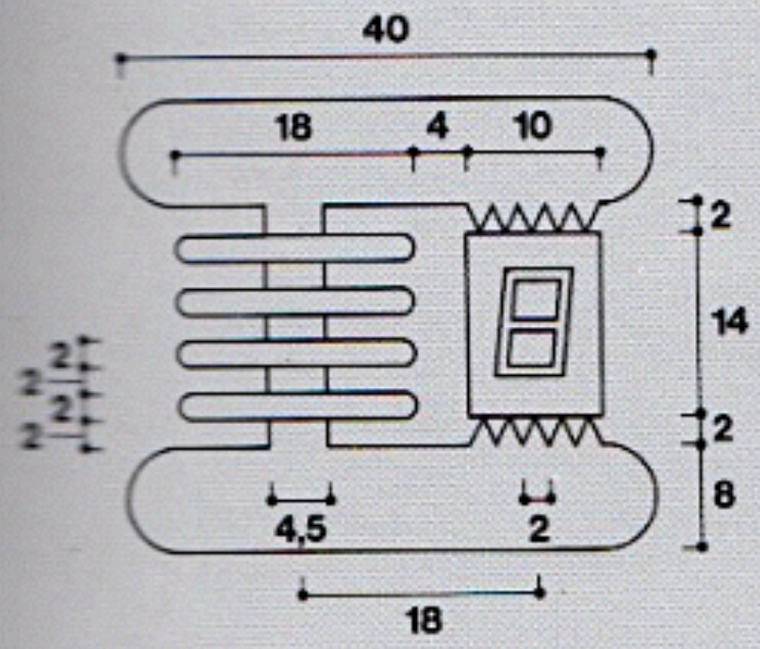


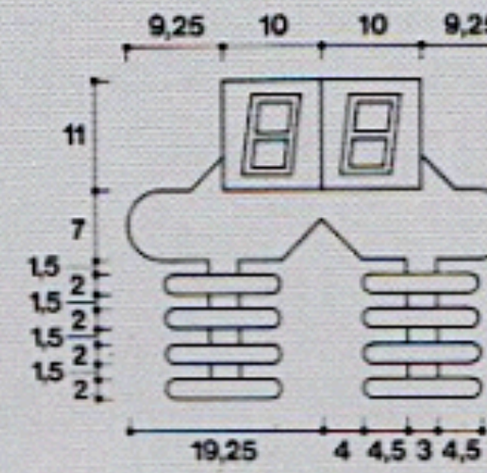
Drawings of the constituent elements of the keyboards. Just like in an alphabet of new characters, despite the smallness, each element is designed in great detail.



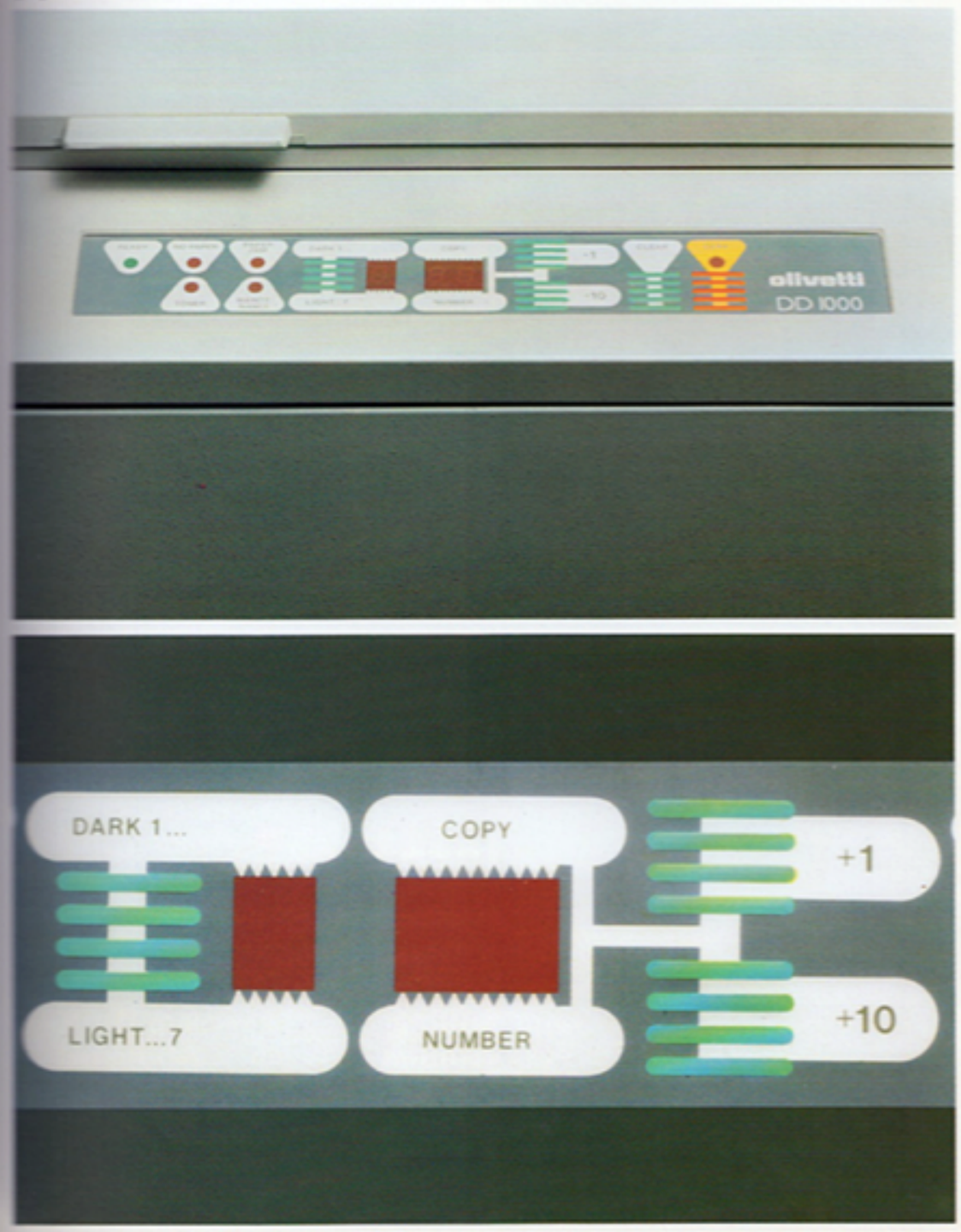




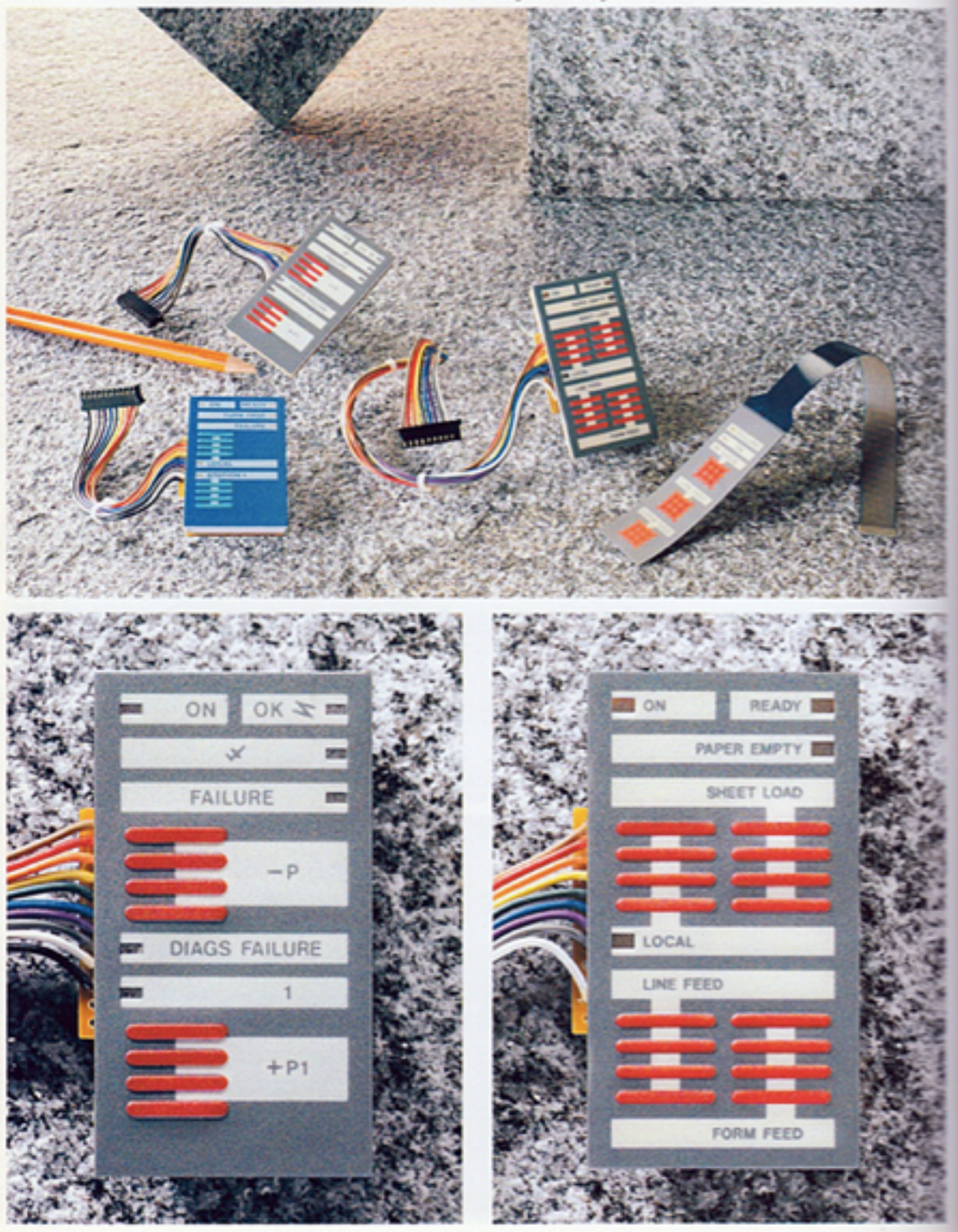




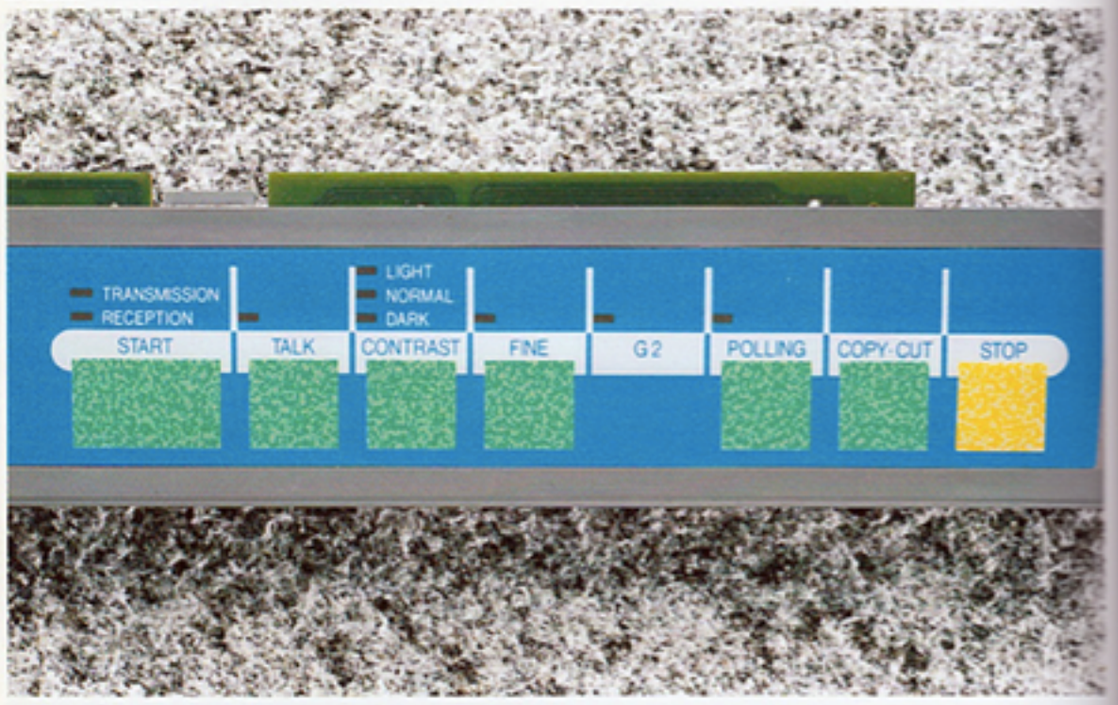
The keyboard of the DD 1000 laser photocopier/printer. Below, the information island formed by a "sequential" key to increase or diminish the brightness, and by two quantity keys (one for units and one for tens of units) which allow the user to set the number of copies desired.



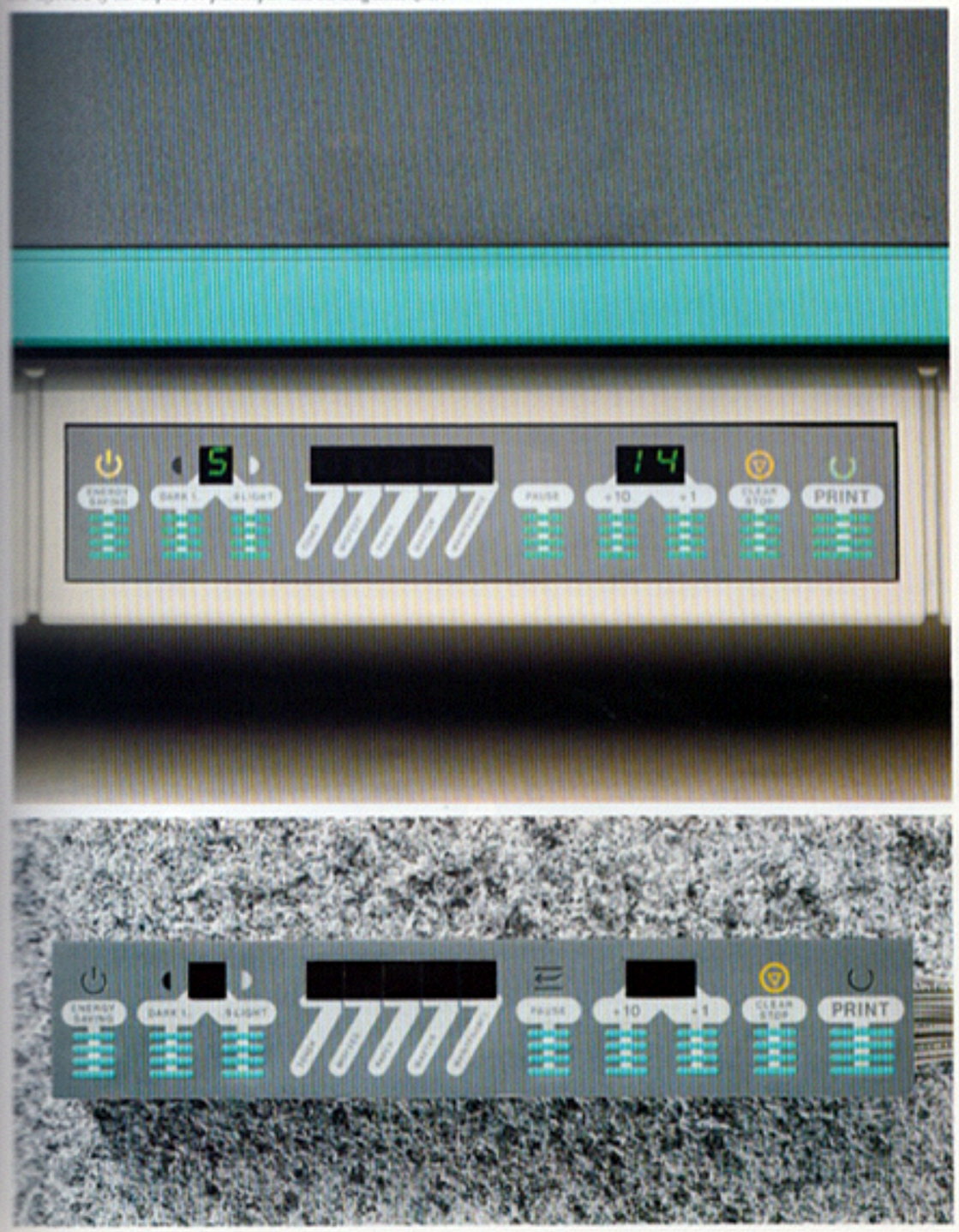
In the latest generation of printers the size of the keyboards has shrunk yet further, thus stimulating new research into materials with which to construct them, and into the methods for manufacturing relief and tactile feedback.



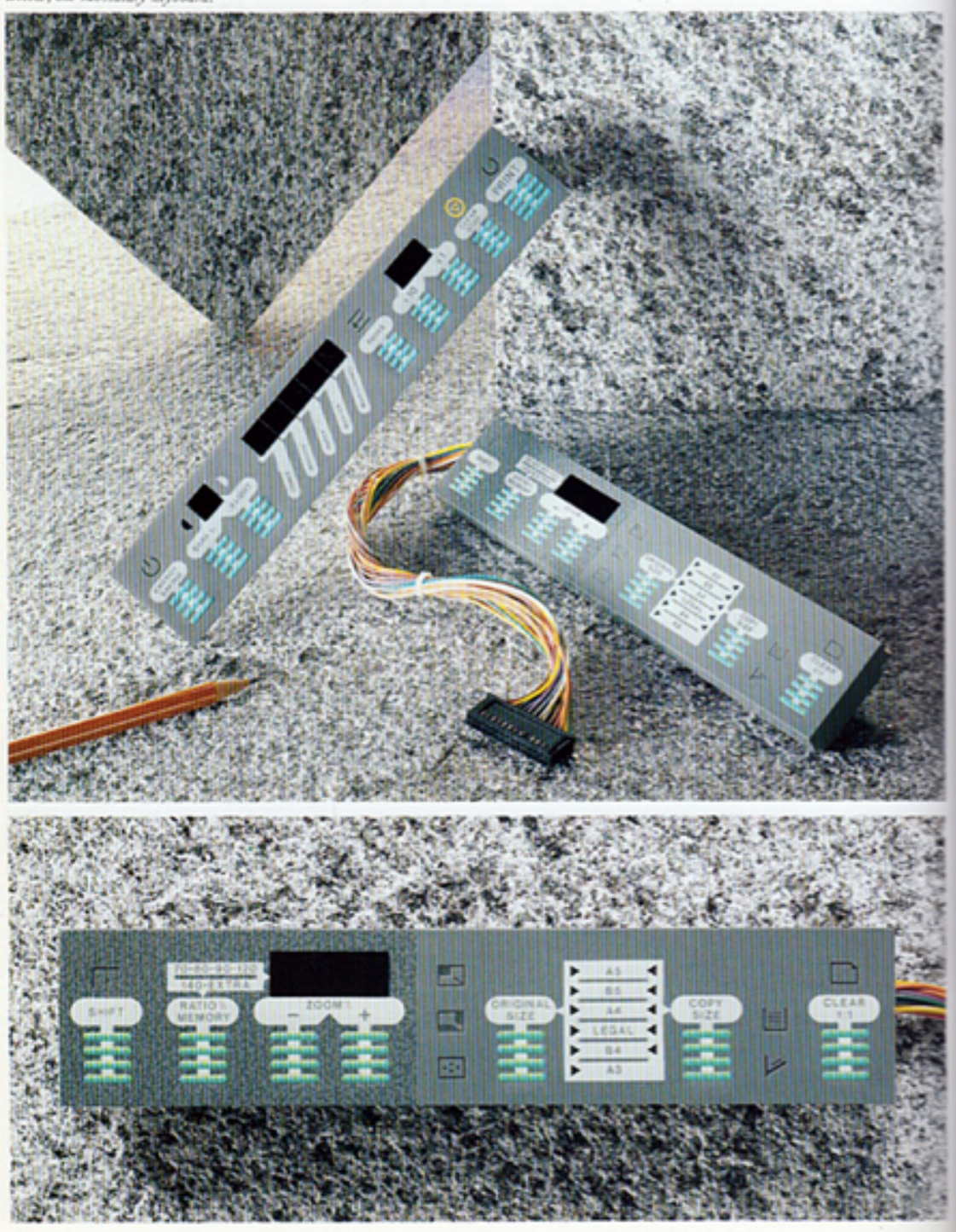
The TLM 340 telecopier and, above, a detail of the function zone of its console.



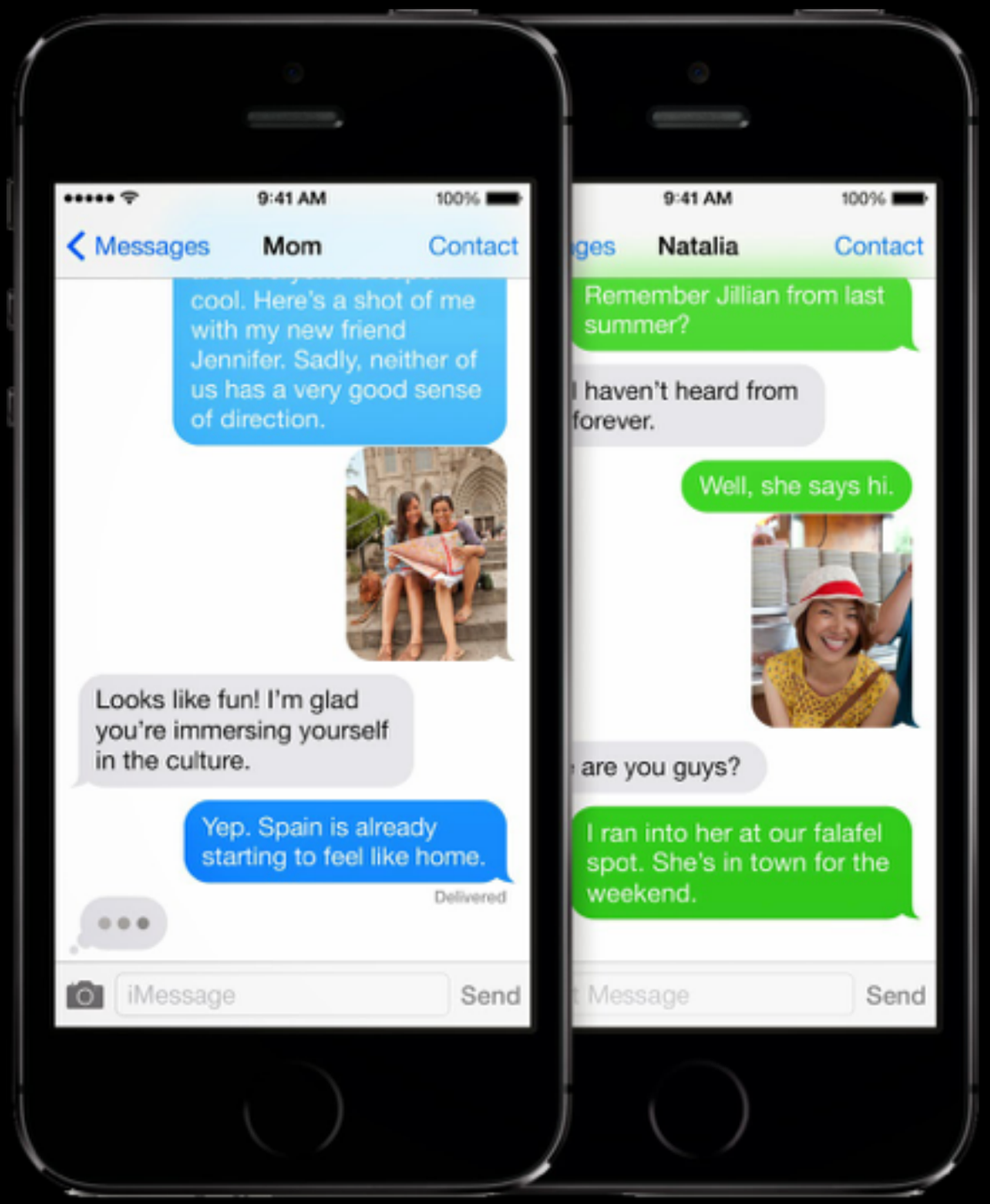
The keyboard of the Copia 900 photocopier and the diagnostic zone.



The principal keyboard and the subsidiary keyboard of the Copia 1150 photocopier.
Below, the subsidiary keyboard.




Carrier
9:45 AM
Messages
Anna Larson
Edit
your name then
Can't wait to catch up
Yea, and I want to hear about your trip
How about 7 tonight?
Sounds great
Send
Q W E R T Y U I O P
A S D F G H J K L
X Z C V B N M
.?123
space
return


9:41 AM
100%
Messages
Mom
Contact
cool. Here's a shot of me with my new friend Jennifer. Sadly, neither of us has a very good sense of direction.
Looks like fun! I'm glad you're immersing yourself in the culture.
Yep. Spain is already starting to feel like home.
Delivered
iMessage
Send
9:41 AM
100%
Natalia
Contact
Remember Jillian from last summer?
haven't heard from forever.
Well, she says hi.
are you guys?
I ran into her at our falafel spot. She's in town for the weekend.
Message
Send

Bloomberg Businessweek

SPECIAL DOUBLE ISSUE June 15 — June 28, 2015 | bloomberg.com

Display Until July 15, 2015
$5.99US $6.99CAN
25>
0 09281 02012 7


```
import datetime

class Issue():
    """TODO write docs here"""
    def __init__(self, **kwargs):
        # TODO: Validate input
        self.__dict__.update(kwargs)

    def publish(self):
        return ('This is the {0.pubdate:%B %d, %Y} issue of {0.title}. ' +
                'It is {0.pages:,} pages long, and ' +
                'costs ${0.price:.5}. '
                'It is about {0.subject}.').format(self)

if __name__ == '__main__':
    bbw = Issue(title='Bloomberg Businessweek',
                price=5.99,
                # That price is only USD;
                # TODO figure out international pricing/currencies
                pages=112,
                pubdate=datetime.datetime(2015, 6, 15),
                subject="code")
    print(bbw.publish())
```

If You Can't Read That, You'd Better Read This

Code: An Essay p.13

Let's Begin

A computer is a clock with benefits. They all work the same, doing second-grade math, one step at a time: Tick, take a number and put it in box one. Tick, take another number, put it in box two. Tick, *operate* (an operation might be addition or subtraction) on those two numbers and put the resulting number in box one. Tick, check if the result is zero, and if it is, go to some other box and follow a new set of instructions.

You, using a pen and paper, can do anything a computer can; you just can't do those things billions of times per second. And those billions of tiny operations add up. They can cause a phone to boop, elevate an elevator, or redirect a missile. That raw speed makes it possible to pull off not one but multiple sleights of hand, card tricks on top of card tricks. Take a bunch of pulses of light reflected from an optical disc, apply some math to unsqueeze them, and copy the resulting pile of expanded impulses into some memory cells—then read from those cells to paint light on the screen. Millions of pulses, 60 times a second. That's how you make the rubes believe they're watching a movie.●

Apple has always made computers; Microsoft used to make only software (and occasional accessory hardware, such as mice and keyboards), but now it's in the hardware business, with Xbox game consoles, Surface tablets, and Lumia phones. Facebook assembles its own computers for its massive data centers.

So many things are computers,● or will be. That includes watches, cameras, air conditioners, cash registers, toilets, toys, airplanes, and movie projectors. Samsung makes computers that look like TVs, and Tesla makes computers with wheels and engines. Some things that aren't yet computers—dental floss, flashlights—will fall eventually.

When you batch process a thousand images in Photoshop or sum numbers in Excel, you're programming, at least a little. When you use computers too much—which is to say a typical amount—they start to change you. I've had Photoshop dreams, Visio dreams, spreadsheet dreams, and Web browser dreams. The dreamscape becomes fluid and can be sorted and restructured. I've had programming dreams where I move text around the screen.

You can make computers do wonderful things, but you need to understand their limits. They're not all-powerful, not conscious in the least. They're fast, but some parts—the processor, the RAM—are faster than others—like the hard drive or the network connection. Making them seem infinite takes a great deal of work from a lot of programmers and a lot of marketers.

The turn-of-last-century British artist William Morris once said you can't have art without resistance in the materials. The computer and its multifarious peripherals are the materials. The code is the art.

How Do You Type An "A"?

Consider what happens when you strike a key on your keyboard. Say a lowercase "a." The keyboard is waiting for you to press a key, or release one; it's constantly scanning to see what keys are pressed down. Hitting the key sends a scancode.

Just as the keyboard is waiting for a key to be pressed, the computer is waiting for a signal from the keyboard. When one comes down the pike, the computer interprets it and passes it farther into its own interior. "Here's what the keyboard just received—do with this what you will."

It's simple now, right? The computer just goes to some table, figures out that the signal corresponds to the letter "a," and puts it on screen. Of course not—too easy. Computers are machines. They don't know what a screen or an "a" are. To put the "a" on the screen, your computer has to pull the image of the "a" out of its memory as part of a font, an "a" made up of lines and circles. It has to take these lines and circles and render them in a little box of pixels in the part of its memory that manages the screen. So far we have at least three representations of one letter: the signal from the keyboard; the version in memory; ▶

● We're all rubes now.

● Which of the following is a computer?

- ○ a. Mugs
- ○ b. Cars
- ○ c. Thermostats
- ○ d. Stereos
- ○ e. ATMs
- ○ f. Cameras
- ○ g. Dinner forks
- ○ h. Tennis rackets
- ○ i. Toilets
- ○ j. All of the above

Answer: j. All of the above

A few of those 1,700 languages are still with us.[24] Cobol, for example, a legendary and much-hated, extremely verbose language that was intimately linked to the "year 2000" problem. As computer ▶

[24] There are a few very big, popular programming languages—and many hundreds of others. Here's an abridged list of important ones, adapted from Wikipedia (which is actually good when it comes to computer things), with some of the most interesting languages noted.

A
A#
A# .NET — Came before B!
A+
A++
A-0 System
ABAP
ABC
ABC Algol
Abset
Absys
ACC
Accent
Ace DASL
ACL2
Action! — Lives inside Adobe Flash
ActionScript
Ada — Created for the U.S. Department of Defense. Good for fighter jets!
Adenine
Agda
Agilent VEE
Agora
Aimms
Alef — New, object-oriented, runs on Java virtual machine
ALF
Algol 58
Algol 60
Algol 68
Algol W
Alice
Alma-0
AmbientTalk
Amiga E
AMOS
AMPL
Apache Flex
APL
App Inventor for Android's visual block language
AppleScript — Knits Mac apps together
Arc
ARexx — Rexx, but for the Amiga computer!
Argus
AspectJ — "Aspect-oriented" Java. Don't ask
Assembly Language — How low can you go? One step up from machine code
Ateji PX
ATS
Autocoder
AutoHotkey — Task-repeater in Windows
Autolt
AutoLisp — A venerable text processor on Unix
Averest
AWK
Axum — Part of AutoCAD—for blueprints

B — Came before C
B
Babbage
Bash — A very, very popular Unix shell
Basic — For beginners, back before Python
batch
bc — Abort, retry, fail? — A tiny calculator
BCPL — All hail
BeanShell — A shell, for Java
Bertrand
BETA
Bigwig
Bistro
BitC
Bliss — Lisp, but by committee
Blue
Bon
Boo
Boomerang
Bourne shell
BPEL — For writing systems until C happened
BREW

C
C — Sound-synthesis programming language
C# — By Microsoft
C++ — Object-oriented C
C--
C/AL
C Shell — Another Unix shell
Caché ObjectScript
Caml
Cayenne
CDuce
Cecil
Cel
Cesil
Ceylon
CFEngine
CFML — "Cold Fusion Markup Language" for making Web pages
Cg
Ch
Chain — Joke language—code looks like recipes
Chapel
Charity
Charm
Chef — Russian space-program flowchart language
Chill
CHIP-8
ChucK — Real-time music programming
CICS
Clik
CL
Claire
Clarion
Clean
Clipper
CList
Clojure — A Lisp that runs on the JVM
CLU
CMS Exec
CMS-2
Cobol — The language everyone (except bankers) loves to hate
Cobra
CODE
CoffeeScript — Simplified JavaScript; yields JavaScript
Cola
ColdC
ColdFusion
Comal
Comit
Common Intermediate Language
Common Lisp
Compass
Component Pascal
Constraint Handling Rules
Converge
Cool
Coq — Lets you write programs as mathematical proofs. Sounds awkward
Coral 66
Corn
CorVision
Cowsel
CPL
Csound — Sound-synthesis programming language
CSP
CUDA — Pre-Cobol data processing from the '50s
Curl
Curry
Cyclone
Cython

D
D — Wants to succeed C++
Dart — Google's proposed JavaScript killer; didn't succeed
DASL
DataFlex
Datalog
Datatrieve
dBase
dc
DCL
DECTPU
Deesel
Delphi
Dibol
Dog — Proposed MIT language for building social networks
Draco
Drakon — Pre-Steve Jobs Apple language (shades of Swift)
Dylan
Dynamo

E
E
E#
Ease
Easytrieve Plus
ECMAScript — Google's answer to Erlang
Edinburgh IMP
EGL
Eiffel
ELAN — Programming language inside a programmer's editor
Elixir
Elm
Emacs Lisp
Emerald
Epigram
EPL
Erlang
es
Escapade
Escher — Created by Ericsson to handle lots of connections
Espol
Esterel
Etoys
Euclid
Euler
Euphoria
EusLisp Robot Programming Language
EXEC 2
Executable UML

F
F
F# — Newish stack-based language
F-Script
Factor
Falcon
Fancy
Fantom — Talk to your database. Tell it how you feel
Faust
Fjölnir
FL
Flavors
Flow-matic
Focal
Focus
Formac
@Formula
Forth — Nice stack
Fortran — Great for plotting trajectories
Fortress
FoxBase
FoxPro
FP
FPr
Franz Lisp
Frege

G
G
G-code
GameMaker Language
GameMonkey Script
GAMS
GAP
GDL
Genie
George — Free variation on Matlab
GLSL
GNU E
GNU Octave
Go
GOAL
Godiva
GOO
Google Apps Script
Gosu
Gotran
GPSS
Grass
Groovy — A scripting language built atop Java
Gödel

H
Hack — Facebook's PHP
HAL/S
Hamilton C shell
Harbour
Hartmann pipelines
Haskell — Good for learning
Haxe
High Level Assembly — Much-lamented language inside of HyperCard
HLSL
Hop
Hope
Hugo
Hume
HyperTalk

I
IBM HAScript
IBM Informix 4GL
IBM RPG
ICI
Icon — Neat text-processing language that inspired Python
Id
IDL
Idris
IMP
Inform — Create text adventures. Get lamp! Try it!
Information Processing Language
Io
Ioke
Iptscrae
ISPF
Iswim

J
J
J#
J++
JADE
Jako
JAL
Janus
JASS
Java — A big language for a big world
JavaFX Script
JavaScript — Runs in almost every Web browser
JCL
JEAN
Join Java
JOSS
Joule
Jovial
Joy
JScript
JScript .NET
Julia — The hot new science, finance, and math language
Jython — Python, but in Java

K
K — Brevity! This program sorts strings: "x@>#;'x"
Kaleidoscope
Karel
Karel++
KEE
Kojo
Kotlin — JVM language made by IDE people
KRC
KRL
Krypton
ksh — Another shell

L
L# .NET
LabVIEW — Plug it into your device
Lagoona
Lansa
Lasso
LaTeX — Make pretty documents
Lava
LC-3
Leda
LIL — Typeset music
LilyPond
Limbo — Assembles your assembly for x86 Intel chips
Limnor
LINC
Lingo
Linoleum
LISA
Lisaac
Lisp — (everything (is a (list) (of lists)))
Lite-C
Lithe
Little b
LiveCode
LiveScript
Logo — Move that turtle, kids
Logtalk
LPC
LSE
LSL
Lua — A friendly embeddable language from Brazil
Lucid
Lustre
Lyapas
Lynx — What assembly becomes

M
M2001
m4
machine code
MAD
MAD/I
Magik
Magma
Make — Software for building software
Maple
Mapper
MARK IV
Mary
MASM — Microsoft Assembly x86
Mathematica — For people with something to prove
Matlab
Max
Maxima — Make live flowcharts that process sound
MDL
Mercury
Mesa
MetaCard
Metafont — For defining fonts!
MetaL
Microcode — What is underneath machine code
MicroScript
MIIS
MillScript
Mimic
Mirah
Miranda
MIVA Script
ML
Moby
Model 204
Modelica — An early object-oriented language
Modula
Modula-2
Modula-3
Mohol
MOO — Hospitals and banks love this language
Mortran
Mouse
MPD
Mumps

N
Napier88
NASM — We'll be ready when you are
Natural
Neko
Nemerle
nesC
NESL
Net.Data
NetLogo — Build little epidemiology simulations
NetRexx
NewLisp
NEWP
Newspeak
NewtonScript — Programming language for the late PDA
NGL
Nial
Nice
Nickle
Nim
Not eXactly C
Not Quite C
NPL
NSIS
Nu
NWScript
NXT-G — What became Java

O
oXML
Oak
Oberon
OBJ2
ObjectLisp
Object Pascal
Object REXX
Objective-C — See: Mac OS and iOS
Objective-J
ObjectLOGO
Obliq
OCaml — French, compiled, functional
occam
occam-π
OmniMark
Onyx
Opa
Opal
OpenCL
OpenEdge ABL
OPL
OPS5
OptimJ
Orc
ORCA/Modula-2
Oriel
Orwell
Oxygene — Way more than 20¢ worth of paradigms
Oz

P
ParaSail
PARI/GP — For the number theorist in your family
Pascal
Pcastl
PCF
PDL
Pearl
PeopleCode
Perl — Made the Web. Still handy
Perl 6 — We'll be ready when you are
PHP — Took over from Perl
Pico
PicoLisp — Like Lisp, but little
Pict
Pike
PIKT
Pilot
pipelines
Pizza
PL-11
PL/0
PL/B
PL/C
PL/I
PL/M
PL/P
PL/SQL
PL360
Planc
Plankalkül — Created by Konrad Zuse in the 1940s
Planner
PLEX — An interactive sketchpad for multimedia. Draws pretty pictures!
Plexil
Plus
POP-11
PostScript
Powerhouse
PowerShell
PPL
Pro*C
Processing — Processing, but for JavaScript
Processing.js
Prograph
PROIV
Prolog — Spock's favorite language
Promal
Promela
Prose modeling language
Protel
ProvideX
Pure
Python — The reason JavaScript has prototypes — A nice language for everyone to know

Q
Q
Qalb
QPL — As object-oriented as it gets
Qt Script
QuakeC

R
R
R++ — A fine new take on Lisp
Racket
Rapid
Rapira
Ratfiv
Ratfor — Fortran, modernized for the 1970s
rc
Rebol — Tiny language by an Amiga person
Red
Redcode
Refal
Reia
Revolution
rex
REXX — For scripting IBM systems
Rlab
RobotC
ROOP
RPL
RSL
RTL/2
Ruby — Object-oriented Rails host
RuneScript
Rust — Up-and-coming systems language

S
S — Statistical language from Bell Labs
S-Lang
S-Plus
S/SL
S2
S3
SA-C
SabreTalk
SAIL
Salsa
SAM76
SAS
SASL
Sather
Sawzall
SBL — Used at Twitter
Scala — Based on Lisp, comp sci
Scheme
Scilab — Make a cat dance (for kids)
Scratch
Script.NET
Sed — Examine many lines of text
Seed7
Self
SenseTalk
SequenceL
SETL
Signal
Simple
Simpol
SimScript
Simula — First "object-oriented" language, 1967
Simulink
Sisal
SLIP — The ultimate glue language
Small
SmallBasic
Smalltalk
SML
Snap! — Does stuff with strings
Snobol
SP/k — Basic in every sense of the word
Span
Spark
Speedcode
SPIN
SPS — The big one, for database access
SQL
SQR — Python without that slow, pesky stack
Squeak
Squirrel
SR
Stackless Python
StarLogo — Makes beeps and boops
Stata
Stateflow
Strand
Subtext
SuperCollider
SuperTalk
Swift — A new player from Apple
Sympl
SyncCharts
SystemVerilog

T
T — Another text-adventure language
T-SQL
TACL
Tacpol
TADS
TAL
Tcl
Tea — Big at early websites. Pronounced "tickle"
TECO — One of the original text editors—very scriptable
Telcomp
TEX — Donald Knuth's amazing typesetting language
TIE
Timber
TMG — Compiler-compiler
TOM
Tom
TopSpeed
TouchDevelop
TTCN — You know it's fast, because turbo
TTM
Turbo C++
Turing
Tutor
TXL
TypeScript — Microsoft's layer atop JavaScript

U
Ubercode
UCSD Pascal
Umple
Unicon — An update to Icon
Uniface
Unity
Unix shell
UnrealScript

V
Vala
VBA — Used to design circuits
VBScript
Verilog — Also used to design circuits
VHDL
Visual Basic
Visual Basic .NET
Visual DataFlex
Visual DialogScript
Visual Fortran
Visual FoxPro
Visual J#
Visual J++
Visual Objects
Visual Prolog
VSXu
Vvvv

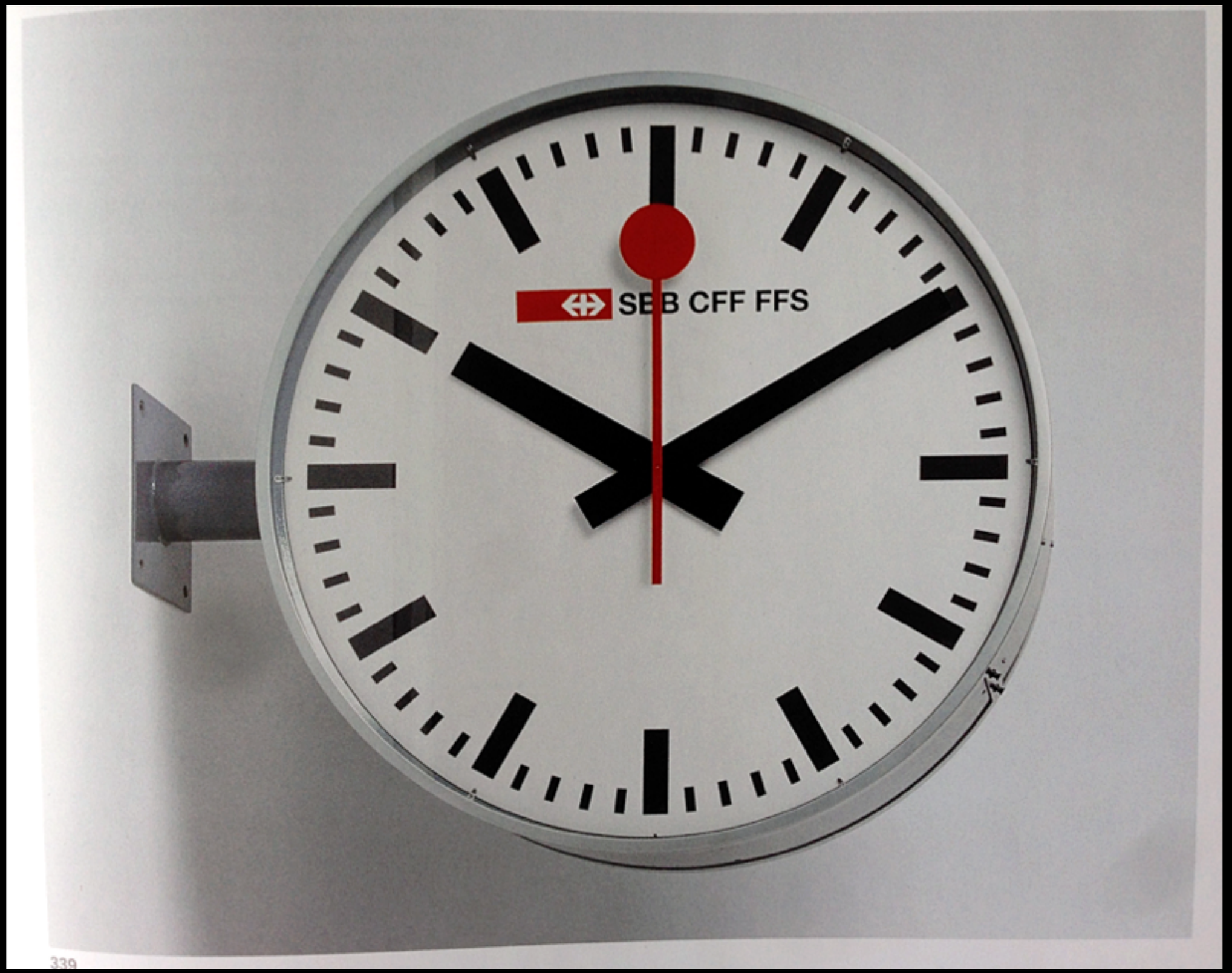
W
WebDNA
WebQL
Winbatch
Windows PowerShell
Wolfram
Wyvern — A data-driven language from the Mathematica people

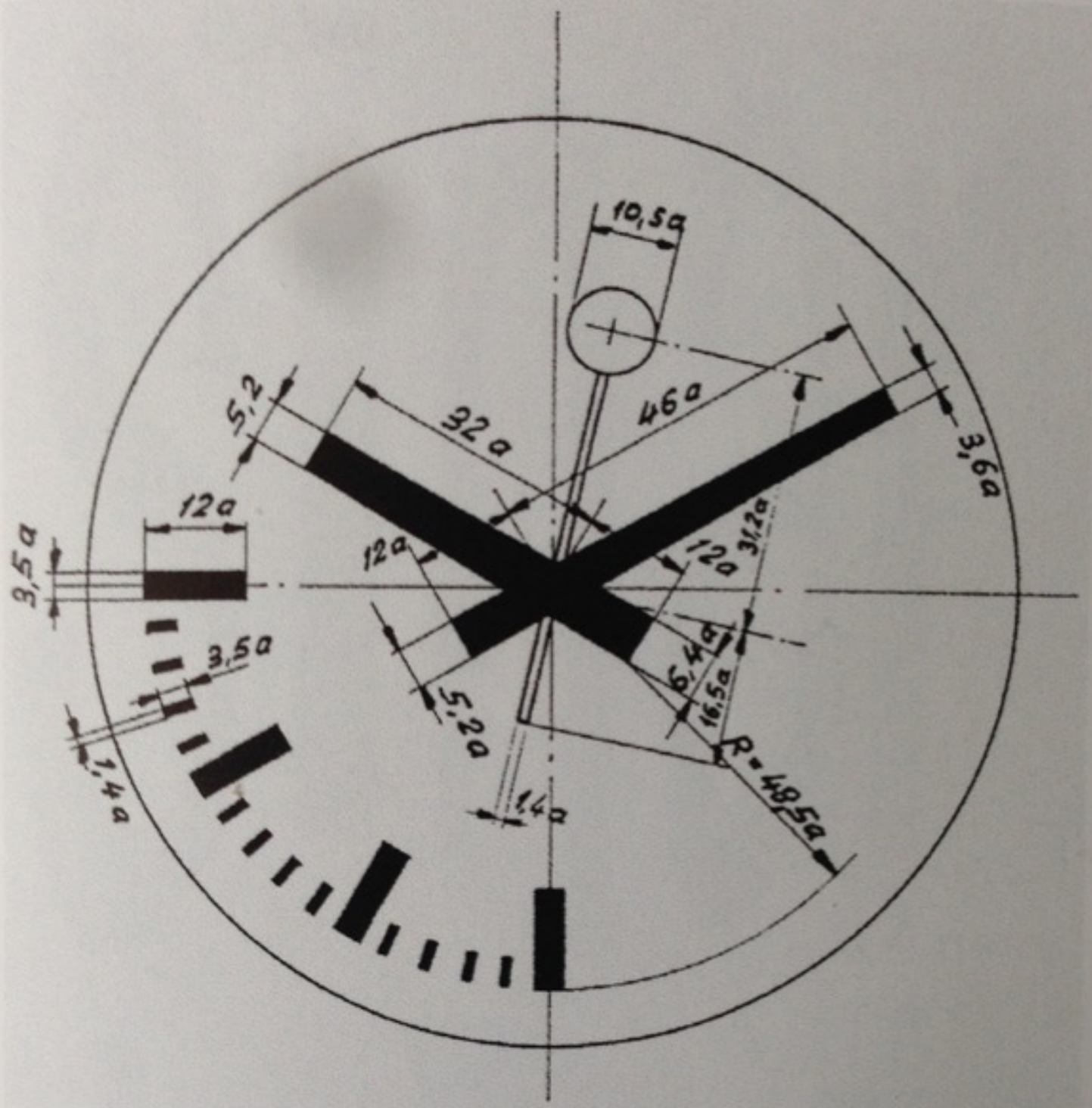
X
X#
X++
X10
XBL
XC
xHarbour
XL
Xojo
XOTcl
XPL
XPL0
Xquery — Access data inside XML
XS B
XSLT — Transform XML data into other things (like Web pages)
Xtend

Y
Yorick — Good for science
YQL

Z
Z notation
Zeno
ZOPL
ZPL — Mostly used in label makers


9


SBB CFF FFS



$$\text{Einheit } a = \frac{\text{Nenn } \phi \text{ (mm)}}{100}$$


iPad
12:07 PM
73%
World Clock
Vancouver
Saturday, 10 November
5°


2 Zürich HB
Winterthur Flughafen
1 C


2 Zürich HB
Winterthur Flughafen
1 C


Zürich HB
Winterthur Flughafen
1 C
STARBUCKS COFFEE


Zürich HB
Winterthur Flughafen
1 C


Zürich HB
Winterthur Flughafen
1 C
STARBUCKS COFFEE

ürich HB
nterthur Flughafen
1 2 2 2
B C D
303 304 305 306 307
OLAT
trani
1 C
STARBUCKS COFFEE


Zentrum
Altstadt
Bus-Information