Preliminary Report on MUNIAC. A newly designed digital computer employing vacuum-tube technology.

John G. Zabolitzky, Gesellschaft f. historische Rechenanlagen e.V., Stäblistrasse 10b, 81477 München www.cray-cyber.org

Abstract

I describe the design process and decision alternatives encountered in building a digital computer today employing technology of the vacuum-tube era. The purpose of this exercise is to achieve a deeper understanding of the problems faced and solved by the computer designers of those days, and to consider the design alternatives they have had.

Contents

Introduction

Someone who is interested in the history of computing, and in particular in the vacuum-tube era down to the details, will inevitably ask himself the question: Why was this design (insert your favorite) done exactly like that ? Of course the general level of knowledge and technology of those times is known, but in what way a specific machine design was affected, why one of several possible alternatives was chosen, is usually not obvious.

A second motivation for this project is the usefulness of having a specific goal to attain, in order to be guided by necessity through a study of all the required details. Just reading descriptions of and stories about historic machines one interpolates very easily over unrealized gaps in understanding, which is impossible if you want to build a functional machine.

Third, being a physicist, I am generally convinced that only the experiment will tell us if we really have understood some matter. Therefore, building a machine today with ancient technology I consider the best way to obtain a solid understanding of those ancient times.

Given this motivation, there is an enormously large field of possible designs open. Some other considerations have to guide the basic design: What size of machine are we talking about ? Do we want to achieve some specific performance goals ? How serious do we take the technology limitation, or where do we exactly draw the line between "acceptable" technology and "unacceptable" technology for this project ? Some external constraints of this sort (which normally, for a commercial product, would be the constraints from the anticipated market) are required.

These were the constraints I selected:

1. Total power required should be compatible with a single-phase standard German outlet, which is 16 A at 230 V, or roughly 4 kVA. This choice is a matter of convenient operation of the machine at essentially any location.
2. Bit-serial operation should not be used. While bit-serial operation would certainly allow a richer functional design with much smaller tube count, I consider bit-serial operation too far remote from the kind of machines I like. This is a statement about personal taste, of course.
3. Memory should be some magnetic core plane(s), to be obtained from old stock. Delay lines, drums, and Williams tubes do not appeal to me, are very difficult to obtain and to maintain. This is equivalent to locating the era of interest towards the end of the 1950s.
4. Cost and effort should be kept reasonable by selecting materials from all available sources, not restricting myself to parts of the original era. While from the point of view of a historically truthful reproduction or from a reconstruction perspective this would not be acceptable, none of these motives are of interest here. So any vacuum tube is acceptable, as are modern diodes, resistors,

capacitors, and metal frames. These are not logically different from the corresponding historic parts but much easier to obtain.
5. Cost and effort minimization has higher priority than performance of the resulting machine.

The constraints severely limit the design space. The first step obviously must be the actual acquisition of the core plane(s), since the entire design will depend upon word size and address size. Since it is close to impossible to find functional core planes dating back to the 1950s I obtained a fully functional Digital Equipment PDP-8E (1971) /1/ with some spare core planes and complete documentation. This allowed me to test the core plane itself, and be aware of the required drive and readout requirements. Therefore, like a PDP-8E or its ancestor, the Control Data CDC-160, the word size is 12 bit, and the address range is 12 bit as well: 4096 or 4k 12-bit words is the MUNIAC memory. While the PDP-8E uses a memory cycle of 1.2 μsec a vacuum tube based design will necessarily be slower than that. The actual cycle time is not defined yet, but will probably be around 10-20 μsec, as will be discussed below.

Given the 12 bit word size and bit-parallel operation, some estimates reveal that the power limitation of 4 kVA is very restrictive. The next step therefore is the design of a low-power and low-cost logic gate and logic family. In other words, at this point we still cannot be sure that this set of requirements is free of contradiction !

Gate-Level Design

Most vacuum-tube computers have been designed with some scheme employing pluggable logic modules, probably the earliest being the IBM 604 calculator of 1948 /2/. One characteristic feature of many historic machines is the existence of a large number of different modules for similar functions. There are two different reasons for this: a) variable number of inputs/outputs combinations; b) variable voltage and drive levels. The latter item is equivalent to saying that the modules do not compose a logic family with universal logic levels, but building blocks from which such logic family is constructed. While this choice certainly reduces tube count, it complicates the design process since many variants need to be designed. I wanted to have a gate design such that the entire machine could be built from a single type of gate.

In the 1940s and early 1950s semiconductor diodes were not available. In the late 1950s they were in wide use. One basic decision in vacuum tube design is the semiconductor-diode gate vs. pure vacuum tube gate /3/. The latter one is ruled out in our case by the 4 kVA restriction, and also the late 1950s placement of MUNIAC technology favors semiconductor diodes. The single gate structure used in MUNIAC therefore is the and-or-invert gate, employing a two-level silicon diode network for the logic and a triode for inversion and amplification.

<<< Fig. 1 >>>
<<< Fig. 2 >>>

Fig. 1 shows the symbolic logic representation, whereas fig. 2 gives the detailed schematics. Without any input connected the voltage at points R and S is about 165 V, since the grid current forces the grid voltage to approximately 0 V. This is taken to correspond to logic true. The triode is therefore conducting, resulting in an anode voltage between 60 V and 100 V, depending upon load at the gate output X. The output value corresponds to logic false, showing the inverting tube action. If we pull any of the inputs A, B, C low (below 100 V), point R will follow that input voltage. Point S will remain at 165 V, however, since it is pulled high due to the D, E, F input section still being high. Thus the AND action of the first-level diodes is exhibited, R = AND (A, B, C), and also the OR action of the second level diode, S = OR (AND (A, B, C) , AND (D, E, F)). When, however, one of the D, E, F inputs is pulled low as well, point S will tend to that lower input voltage as well. The clamping diode to 100 V will limit this low voltage to 100 V. The grid voltage will then be –24 V cutting off the tube. The output voltage at X will then approach 230 V, becoming logically true as a consequence of the diode gates being false now.

The specific values for the components and power supplies result in this data for the gate:

U_{OHmin} = 170 V	U_{IHmin} = 140 V	U_{Htyp} = 230 V
U_{OLmax} = 110 V	U_{ILmax} = 120 V	U_{Ltyp} = 60 - 100 V

valid for a fanout between 0 and 5. One output may be connected to up to a maximum of five inputs. This limitation is due to the high impedance of the inverters. This high impedance also results in a dependence of the actual output level upon the load, where the actual loading due to an input depends upon the logic levels of all other inputs to the same gate. In other words, the voltage present at a gate output changes dynamically, even if the logic remains the same. There are significant variations in the low output level, since the switched-on inverter tube represents an impedance of about 6-8 kOhm. The high output level is stable, since a high output always results in all connected diodes to be reverse biased.

This observation is the reason for the clamping diode to +100V supply. The actual input to the inverter is thus clamped at +100V, and variation of the gate inputs below +100V will not appear at the grid. This is very important for the speed of the circuit. Any variation at the two-level diode gate output point S will be transferred to the grid directly by the 22 pF capacitor. If that capacitor were absent, the effective grid capacitance would need to be charged up via the 120 kOhm resistor, which would be rather slow. Therefore, no spurious voltage changes – even if outside the nominal logic transition – can be allowed at the diode gates output S. Unfortunately, due to the finite capacitance of the reverse-biased diodes, there will still some voltage jump be transferred to the diode gate output even in spite of clamping. This is what limits the grid capacitor to 22pF, and thereby limits the gate speed and therefore the machine cycle time.

From this basic gate any functionality can be built. A latch is built from two gates and a master-slave flip-flop is built from two latches, or four gates.

<<< Fig. 3 >>>
<<< Fig. 4 >>>

The latch circuit of fig. 3 determines the minimum width of clock pulses. In the fully loaded outputs case, where we have maximum capacitance and therefore slowest circuit operation, the minimum clock pulse width is about 3 μsec for this latch to capture the data reliably under any circumstances, with sufficient margin.

A counter bit does not require any more of these gates, but just wider ANDs and ORs.

<<< Fig. 5 >>>

The clocking of this counter is typical for MUNIAC, and similar clocking schemes are used for all registers. A four-phase non-overlapping clock is used. The first phase clears the master (used for the counter only in the case of parallel load), the second phase sets the master if input logic is true, the third phase moves the master data into the slave, and the fourth phase sets up the control lines for the next cycle, i.e. loads the instruction decode latches. Since about 3 μsec pulse width are needed, the MUNIAC cycle time will be approximately 4 * (3 + 1) = 16 μsec or 64 kHz, where one μsec is used to make sure the nonoverlapping of clock pulses.

One important decision is the selection of tube type. Dual miniature triodes within one envelope are the appropriate choice. These were certainly available in the late 1950s, and require least space and power. The tube type selected should have high current capability (for large fan-out) and should be cut-off at a grid voltage of around –10V for sufficient margin in logic levels. The specific type of 6BQ7A was selected since it has /4/ a maximum cathode current of 20 mA, anode power rating of 2 W, cuts off at –8V for the supply voltage of 230 V selected, has low capacitance, and could be obtained cheaply in large quantities from US military surplus. Heater power is 2.5 W for the tube, i.e. for both sections. This tube has been designed as a high-frequency UHF tube, not as a computer tube. Apart from the possible danger of cathode interface (some older tube designs exhibit cathode emittance problems when kept in cut-off state for too long a time /5/) this is not very relevant; the low capacity resulting from the UHF-design is helpful, though. The total power consumption per gate (counting the entire and-or-invert structure as one gate) is 3 W on the average. For a specific gate the power consumed depends upon the fanout.

The diodes were selected as silicon type BAV21, since this is a very inexpensive diode (of current production) capable of withstanding the high reverse voltages of around 200 V. Since this diode is the most frequent of all parts used, its price is particularly important. For some realism in physical appearance, only axially leaded parts are used, no radial leaded parts or even surface-mount, which is very remote in time from the era of study here.

The resistors used are modern 5% 1.4 W or 1% 0.6 W metal film types. The latter choice might be questionable since 5% tolerance was more abundant and quite less expensive in the 1950s. However, the gate design does allow for 5% tolerance.

The mechanical composition of the tube module was also chosen from currently available materials. In order to limit the amount of time required for assembly a single Eurocard (100 mm x 160 mm) single-layer printed circuit board was designed and fabricated. Such single-layer PCBs were available towards the end of the 1950s. The line width used might be somewhat smaller than what was used then, but not in an excessive way. Two tubes with a total of four triodes are mounted on one module, i.e. four and-or-invert gates. The layout of the PCB is such that by stuffing options any distribution of 32 inputs over the four gates can be realized. This gives rise to about 40 different module types, all realized from one and the same basic gate design, and one and the same PCB. The tubes themselves are mounted on a small second PCB which in turn is mounted at right angle to the main Eurocard PCB, in order to conserve space.

<<< Fig. 6 >>>

One of the reasons of using the Eurocard scheme is the availability of standard connectors, 19" card cages, blower assemblies and 19" racks. No special design or manufacture for these things is required therefore, significantly contributing to lowering the cost of the project. Each card also contains four small neon tubes for optical display of each gate state, and testpoints for these outputs and also for the inverter inputs, i.e. diode-network outputs.

<<< Fig. 7 >>>

Historic machine designs use cathode followers for stabilization of gate output levels /6/. A cathode follower like fig. 8 would present a very low impedance to the output. This would make output level rather independent of dynamic loading conditions. This would limit the undesirable spurious voltage jumps at the gate inputs, and therefore enable use of a larger grid capacitor, speeding up the entire design. However, the 4 kVA total power limitation rules out the use of extra cathode followers.

<<< Fig. 8 >>>

The alternative pure tube gates avoiding any silicon diodes would require much larger a tube count for the same number of logic functions. Typically, one would require one tube section per input as opposed to one tube section per output, see fig. 9. Using pentagrid tubes like the 6BE6 /6/ would require one tube per two inputs, exactly the same tube count. These choices again are ruled out by the total power ceiling, as well as cost of the machine.

<<< Fig. 9 >>>

Apart from the logic gate discussed above, three more module types not incorporating any logic are required: clock drivers, magnetic core drivers, and core sense amplifiers. Clock drivers are constructed using the logic PCB over again, the changes being a different tube type, and no diode gating in front of the inverter (tube). Tube type selected is 6CW5, a 14 W miniature power pentode /4/ providing a fanout of 30 due to a maximum current of 120 mA. One pluggable module contains two clock drivers.

The core drivers will be constructed from tube type 40KG6 capable of carrying 1A of current. The detail design has not been done yet, same for the sense amplifiers.

The power supply involves linear regulation with parallel 6KG6 tubes, the reference voltage being supplied by cold-gas glow tubes, a very standard textbook design /7/. There exist exactly two power supplies, +230 V at 8 A and –210 V at 2 A. The +100V is generated from the +230V. Heaters are powered from 24V transformers, using four 6.3V filaments in series, at a slight underheating. Slow-start for the heaters is provided. Total heater power is approximately 1.5 kVA.

Architecture

The basic architecture is the classical von Neumann single-address accumulator-based design /8/. A block diagram is given in fig. 10. The only CPU registers visible to the programmer are the accumulator and the program counter. I/O is done under CPU control via one input register and one output register, no provision is made for a channel (DMA) or peripheral processor. Operator control is via a 12 bit switch bank and various control switches. A Nixie tube display of four octal digits (12 bit) is provided, which can be switched to a number of registers. The highest 64 memory addresses are implemented as diode ROM, not core memory, which allows for a bootloader from the I/O device, or hardware maintenance routines. The diode ROM is on one single pluggable Eurocard module and may easily be exchanged /9/.

<<< Fig. 10 >>>

The I/O device is an electromechanical Siemens 5-channel teletype with built-in paper tape punch and read stations, circa 1958. I/O speed is 50 Baud or 10 5-bit characters per second. No provision is made for any other I/O device. The I/O registers therefore are shift registers for transformation of parallel to serial data.

Machine instructions are either 12 bit or 24 bit. No provisions for indirect or relative addressing, indexing or subroutine linkage are provided. All of these are to be performed by modification of the stored program. Thus, indexing or indirect addressing will be performed by storing the effective address into the address field of a subsequent instruction. Similarly, return addresses for subroutines are to be stored explicitly by the calling routine to prearranged memory locations. One possible and historically used scheme is to store the return address to "routine address", and then jump to "routine address + 1". This "very" von Neumann design was chosen because of its contrast to modern, more Harvard-like instruction repertoire styles.

Instructions not requiring any operands are 12 bit, like e.g. "shift left". If an operand is required this may be either in the form of 12 bit immediate data or 12 bit address. Therefore these latter instructions are 24 bit wide. This scheme corresponds to 12 bit opcode plus optional 12 bit address or immediate data. For a small machine like MUNIAC the 12 bit opcode represents a very long instruction word (VLIW) /10/. There is only little decoding done; essentially, there are only five different operations, selected by bit 11 and bit 10 of the opcode, and some decoding of data source bits in order to distinguish the "Operate no operand" instruction:

Operate immediate	12 bit opcode + 12 bit data
Operate address	12 bit opcode + 12 bit address
Operate no operand	12 bit opcode
Store condition address	12 bit opcode + 12 bit address
Jump condition address	12 bit opcode + 12 bit address

The further bits of the opcode word are assigned directly to the control lines of the various registers, multiplexers, ALU etc., or alternatively to the condition code multiplexers. Conditional store may be efficiently used to replace conditional jumps in some cases. The "operate no operand" is useful for instructions not requiring any data, like shift/rotate operations, clear accumulator or carry bit manipulations.

One important feature is a versatile carry flip-flop. Because of the small word size it can be imagined that multiple-word arithmetic might be frequent. Therefore, four of the 12 opcode bits are permanently assigned to carry handling. For the operate instructions, the bit assignment is

11-10	basic opcode, see above
9 – 8	carry operation: clear, hold, load from ALU, set
7 – 6	ALU carry input select: zero, sign, carry, one
5 – 4	ALU data input select: none, I/O input, memory data, memory data inverted
3 – 0	ALU operation: clear, load, add, shift left, shift right, and, or, nor, xor, nop, output I/O

The "none" selection from the field in bits 5 and 4 results in the "Operate no operand" instruction as a modification of the normal operate instruction.

For store/jump conditional, the bit assignment is

11 – 10	basic opcode, see above
9 – 0	condition select from various accumulator, carry, or I/O conditions

Various more conventional instructions may be obtained by suitable combinations of the opcode bits. A subtract, for example, is obtained by using add, carry input select one, data input select inverted, for a two's complement subtract. The number system (one's complement or two's complement) is not fixed by the instruction set, it is a decision of the programmer what number system he wants to use. A rotate left is obtained from shift left, carry input select sign. Similarly, other shift logical/arithmetic or rotate versions may be obtained.

While this instruction repertoire may appear somewhat strange at first, it is related to more modern ideas about efficient use of hardware by explicitly making use of parallelism within one instruction /10/. Certainly a 12-bit opcode appears rather large for a 4096 word main memory. The classical alternative as implemented in the CDC-160 or PDP-8 would have been a few bits opcode plus a few bits of address, working together to produce 12-bit effective addresses by indirect or relative addressing. The apparent uneconomic use of opcode bits in MUNIAC is offset by the use of parallelism within each instruction, i.e. hopefully fewer instructions are needed in order to accomplish the same task. Conditional store is an example for this: in place of the two instructions conditional jump, unconditional store as more usual only one conditional store instruction is needed. The same holds true for sequences of arithmetic and carry manipulation instructions, where again two conventional instructions may be merged into one MUNIAC instruction. This latter case is frequent in arithmetic routines for multiple wordlength, multiply, etc.

This VLIW scheme of programming is also related to direct microprogramming of the CPU. One could say that there is essentially no machine language in MUNIAC, but the instruction repertoire constitutes the microcode for MUNIAC, and all MUNIAC programs are microcode sequences.

It is quite apparent that no connection with any historic machine design is sought here. On the contrary, the philosophy behind this implementation is to investigate another alternative than chosen by the historic designers. This alternative would have been available in principle to the 1950's engineers, but was not implemented at that time to my knowledge. Implementation of and later programming for this architecture will reveal the relative merits and/or disadvantages. No detailed studies have been carried out yet.

The execution time of instructions is tied to the machine cycle time discussed above. The core memory will need to be controlled from logic, therefore one logic cycle is equivalent to either a core memory read or a core memory write cycle. Since core memory read cycles are destructive, each read has to be followed by a write. A core memory write has to be preceded by a read in order to clear the zero bits. Actual execution is overlapped with the next instruction fetch for the operate instructions. Instruction timing therefore is given simply by the core memory cycle count

Operate no operand	2 cycles	read+write instruction word
Operate immediate	4 cycles	read+write instruction+data word
Operate address	6 cycles	read+write instruction+address+data word
Store conditional	6 cycles	read+write instruction+address+data word
Jump conditional	4 cycles	read+write instruction+address

However, for convenience in the control logic one extra cycle gets added to the 2 and 4 cycle instructions. MUNIAC instructions therefore execute in between 3 to 6 cycles, resulting in approximately 10 KIPS (kilo instructions per second) or 0.01 MIPS.

The entire CPU requires a total of 260 modules, containing four gates each. The total count of and-or-invert gates therefore is 1040. In addition, 30 clock driver modules with a total of 60 clock drivers are required. Total tube count is approximately 600.

The modules are arranged within standard 19" card cages of three height units. This card cage holds 14 modules. Since 12 logic modules plus two clock driver modules are usually required for a 12-bit register this is a very convenient packaging: one card cage is one register or one block of the 12 bit data path. The modules plug into wire-wrap connectors. The back side of the 19" rack therefore forms a plane of wire wrap posts, the backplane wiring is a wire-wrap mat. There is a total of three 19" racks used: two for the CPU and I/O (fig. 7), one for the core memory including drivers and sense amplifiers.

Current State

At the present point in time the logic design is completed. All required parts are manufactured and/or procured. About 50 % of the logic modules is assembled, and 20 % of the backplane wiring is completed, employing wire-wrap technology. The two presently existing 19" racks eventually comprising the CPU are shown in fig. 7. The program counter and clock generators are functionally tested. The logic design was carried out employing modern CAD tools for reason of minimization of design time.

On the author

The author currently is managing director of ICOS Vision Systems GmbH, a company involved in research and development for cutting-edge machine-vision systems, involving modern high-performance digital signal processors. The author has formerly been a Professor of Physics and Fellow of the Supercomputer Institute, University of Minnesota, Minneapolis. He obtained his PhD in theoretical Physics at Ruhr-University, Bochum, Germany, in 1972, and is living in Munich, Germany.

References

1. C. Gordon Bell, J. Craig Mudge, John E. McNamara, Computer Engineering, Digital Press 1978, p. 175 ff
2. Charles J. Bashe, Lyle R. Johnson, John H. Palmer, Emerson W. Pugh, IBM's early computers, MIT press 1986, p. 58 ff
3. R. K. Richards, Digital Computer Components and Circuits, van Nostrand, 1957
4. General Electric, Essential Characteristics, reprinted by Antique Electronic Supply, Tempe, AZ
5. G. Haas, Grundlagen und Bauelemente elektronischer Ziffern-Rechenmaschinen, Philips technische Bibliothek, 1961, p. 91 ff
6. IBM 604 Electronic Calculating Punch, Customer Engineering Manual of Instruction, IBM 1958; see also refs. 2 and 3
7. K. Steimel, Elektronische Speisegeräte, Franzis Verlag München, 1957
8. A.W. Burks, H.H. Goldstine, J. von Neumann, Preliminary Discussion of the Logical Design of an Electronic Computing Instrument, 1946, reprinted in The Origins of Digital Computers, ed. Brian Randell, Springer Verlag, 1982
9. The ROM was added on a suggestion by Hans Franke, personal communication, Munich, 1999
10. G. S. Almasi, A. Gottlieb, Highly Parallel Computing, Benjamin 1994, p. 622 ff

Figure Captions

Fig. 1. Basic and-or-invert gate, logic representation. The number of inputs is variable, i.e. many realizations exist.

Fig. 2. Basic and-or-invert gate, schematic diagram.

Fig. 3. Latch built from two gates

Fig. 4. Master-Slave Flip-Flop built from two latches or four gates.

Fig. 5. One bit of a binary counter, requiring four gates

Fig. 6. Photograph of pluggable four-gate module

Fig. 7. Current State of MUNIAC assembly, two 19" racks

Fig. 8. Cathode follower, as it is not used in MUNIAC

Fig. 9. NOR gate built from two triodes, as it is not used in MUNIAC

Fig. 10. MUNIAC block diagram

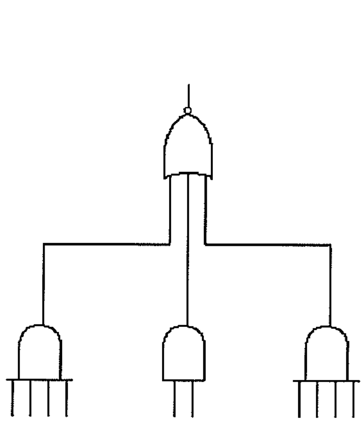

Fig. 1

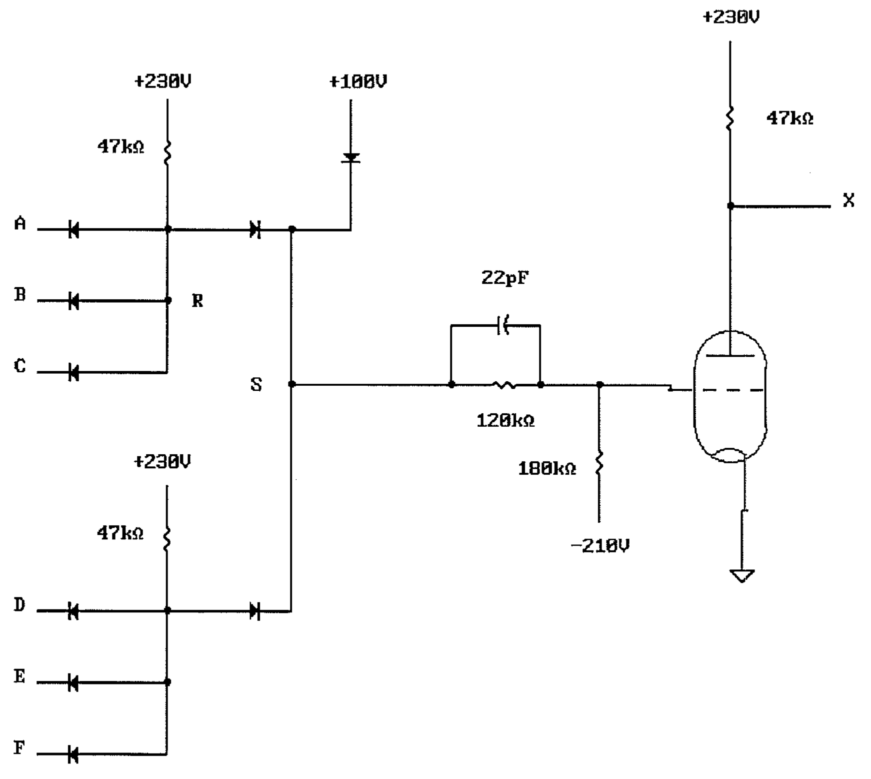
+230V
47kΩ
+100V
A
B
R
C
S
+230V
47kΩ
D
E
F
22pF
120kΩ
100kΩ
-210V
+230V
47kΩ
X

Fig. 2

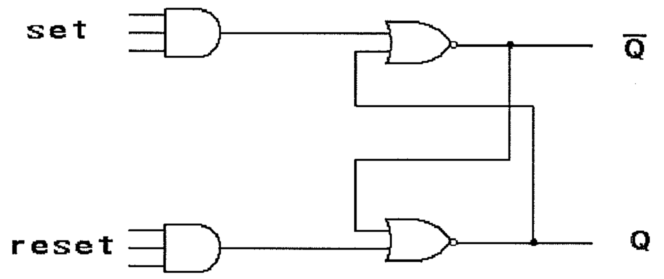

Fig. 3

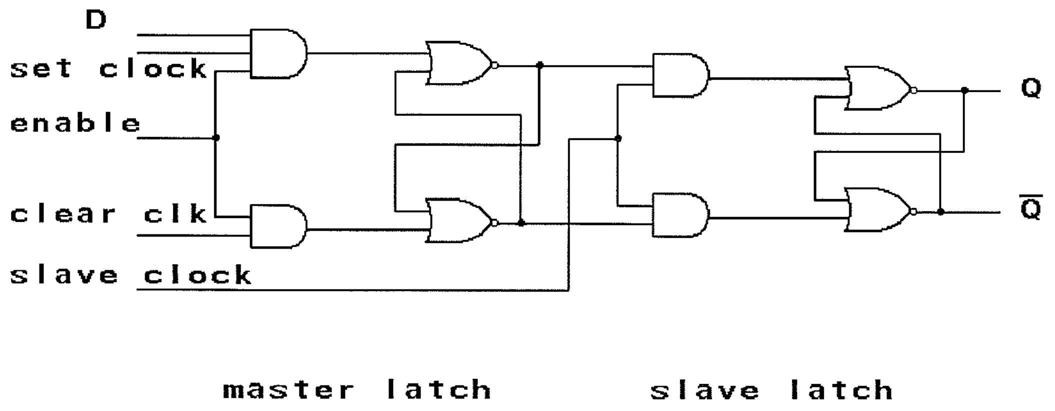
D
set clock
enable
clear clk
slave clock
Q
$\overline{Q}$
master latch
slave latch

Fig. 4

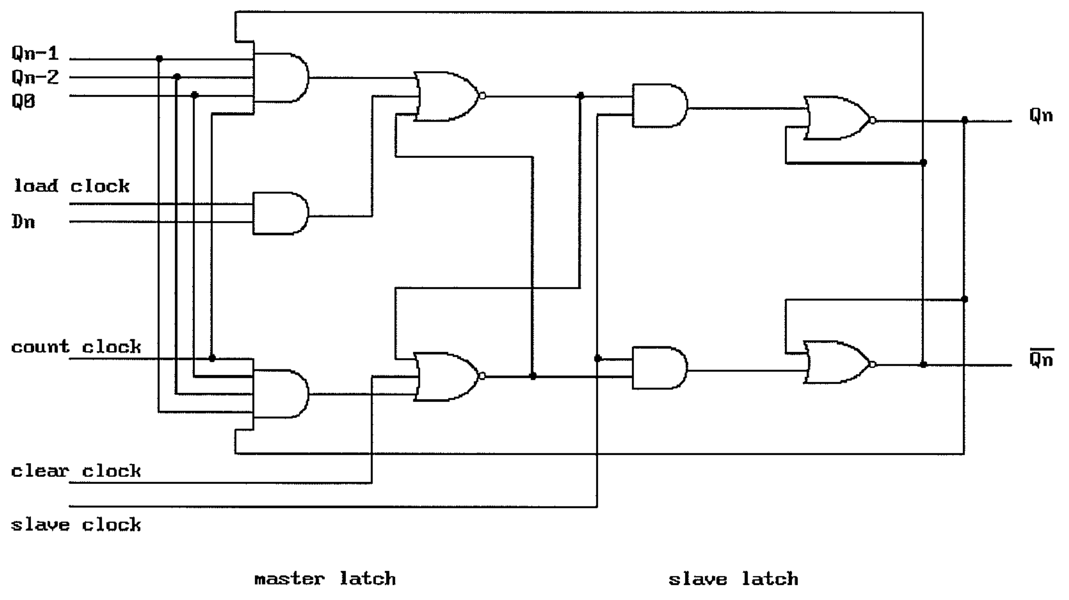
Qn-1
Qn-2
Q0
load clock
Dn
count clock
clear clock
slave clock
Qn
$\overline{Qn}$
master latch
slave latch

Fig. 5

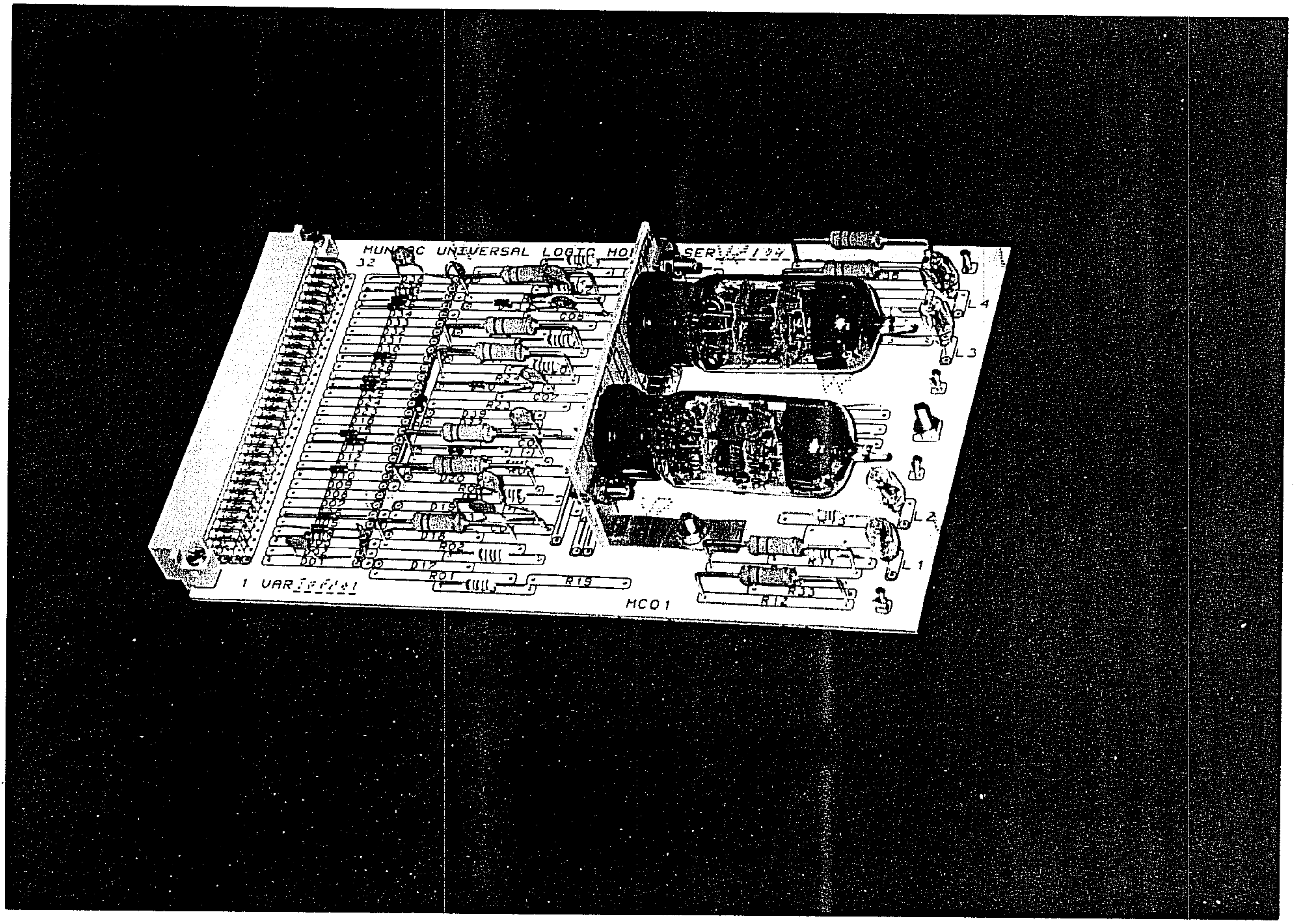
MUNTAC UNIVERSAL LOGIC MO
SER
J2
L 4
L 3
L 2
L 1
1 VAR
MC01
R19
R01
R02
R12

Fig. 6

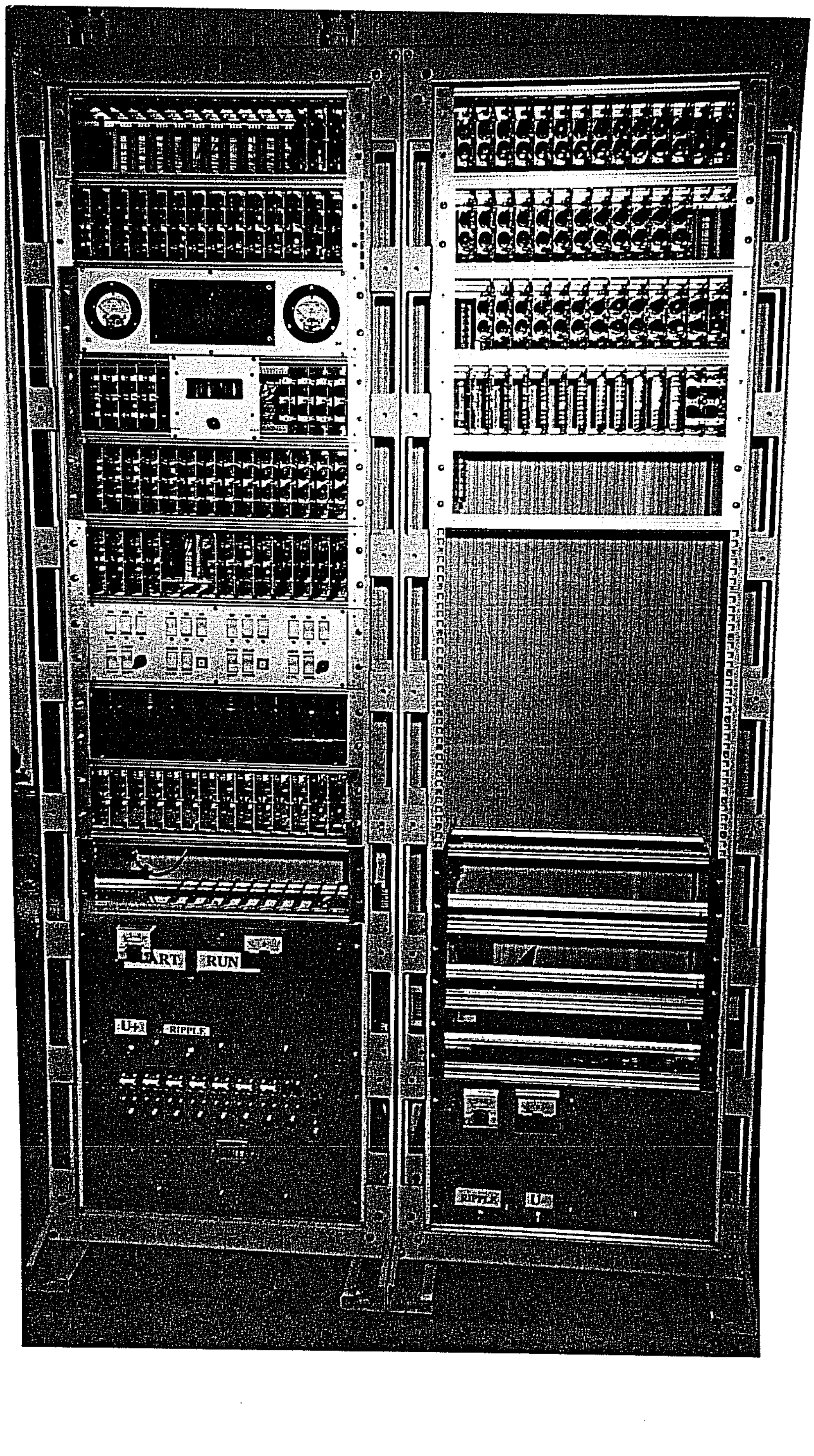

Fig 7

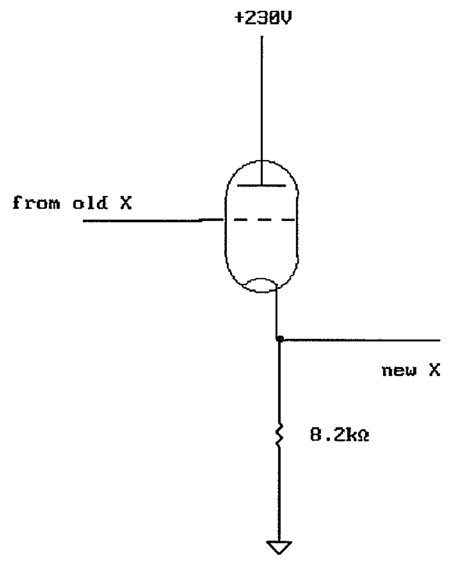

Fig. 8

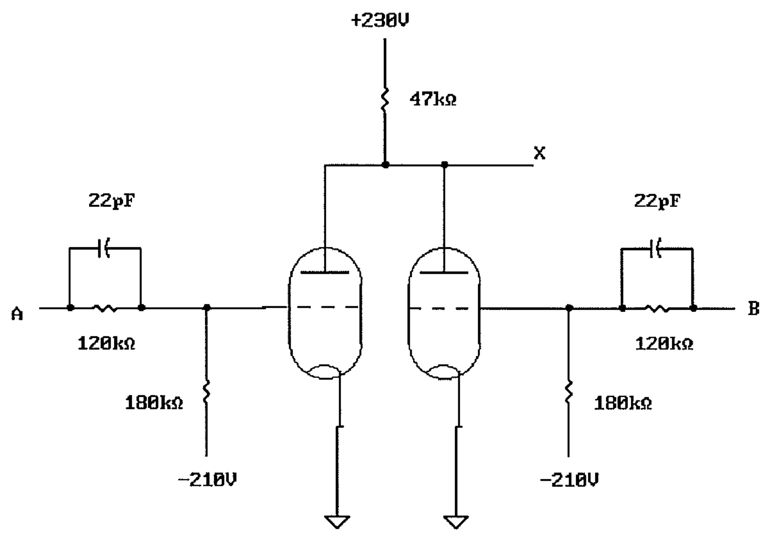

Fig. 9

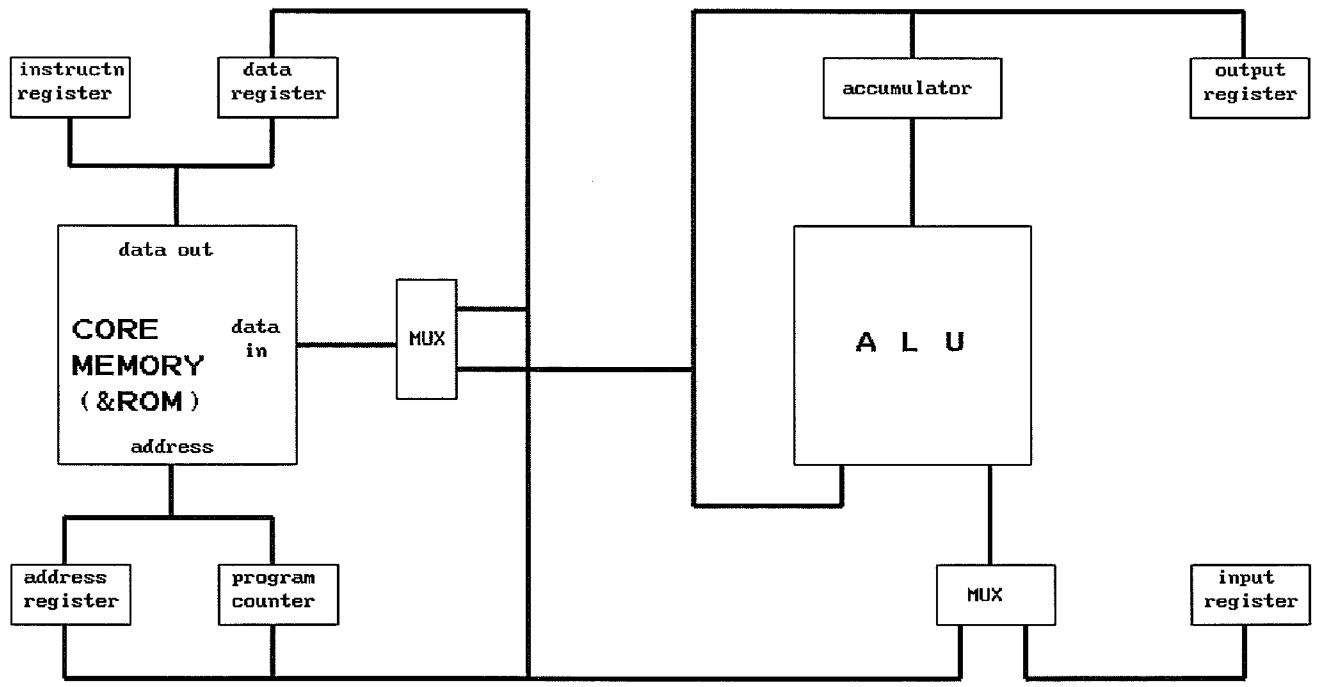
instructn register
data register
accumulator
output register
data out
CORE MEMORY (&ROM)
data in
address
MUX
A L U
address register
program counter
MUX
input register

Fig. 10