ISSUE	DRAWING NO.

REVISIONS

ISSUE	DESCRIPTION	DWN.	E.C.O.	DATE	APPD.

QANTEL CORPORATION
3474 INVESTMENT BLVD.
HAYWARD, CALIF. 94545

RELEASE DATE		
DWN.	DR	12-1-71
CKD.	BL	12-1-71
PROJ.	BF	12-1-71
MFG.	[illegible]	12-1-71
ENG.	[illegible]	12-7-71

SPECIFICATION

QANTEL CENTRAL PROCESSING UNIT (CPU)
DESIGN SPEC.

SHEET	DRAWING NO.	ISSUE
1 OF 13	A3-0300	A

1.0 INTRODUCTION

1.1 This specification is provided to acquaint the reader with the functional operation of the CPU. The functional description of the unit is presented in descending levels of explanation beginning with a brief and general description of the micro-programmed computer concept, followed by an overall block diagram-type description of the QANTEL V Processor; a listing of the complete micro-instruction set and finally, a tabular specification of the processors capacity.

2.0 MICROPROGRAMMED COMPUTERS

2.1 The QANTEL V CPU is a micro-programmed computer employing a read-only memory and associated registers for the control unit. Within the micro-programmed computer, all macro-operations are performed by specific algorithms composed of micro-instructions selected during computer manufacture. During processor operation, the control unit executes a sequence of micro-instructions (an algorithm) stored in an addressable portion of a read-only memory. The micro-instructions making up the algorithm for any particular macro-instruction are sequentially produced at the output of the read-only memory, where they are decoded and used to perform simple processor functions,-such as moving data between registers, etc. As the execution of each micro-instruction is completed, the subsequent instruction is brought out of the read-only memory and decoded. This instruction is then executed, and so on, until the algorithm (and the macro-instruction) is completed.

2.2 All micro-instructions contained in the read-only memory are part of a standard micro-instruction set, but are arranged in a specific and different sequence to accomplish each macro-instruction. A particular micro-program (algorithm or sequence of micro-instructions) in the read-Only memory is addressed during the basic fetch cycle by using the macro-instruction operation code. By this method, each different macro-instruction operation code addresses a specific portion of the read-only memory containing the correct algorithm (or micro-program) required to perform the associated macro-instruction.

2.3 In summary, the operation code of the macro-instruction is used to address the associated sequence of micro-instruction residing in the read-only memory. The sequence of micro-instructions (micro-program) is then executed in a step-by-step fashion until the macro-instruction has been completed. At this time, the next macro-instruction is fetched and the operation code is used to address the corresponding micro-program or algorithm in the read-only memory required to perform that instruction.

3.0 PROCESSOR ORGANIZATION

3.1 General

3.2 In order to understand the details of processor operation, it is first necessary to study the organization of the processor on a block diagram level as shown in Figure 1. The processor may be divided into six main functional areas. These areas include:

a. The control unit (read-only memory, associated registers, and micro-instruction decoder).

b. Direct memory access and priority logic.

c. The main memory.

d. The data and program registers.

e. The arithmetic - logic unit.

f. The data bus and controllers.

3.3 Control Unit

3.3.1 The control unit governs the overall operation of the system, and is comprised of the Q register, the Z register, the read-only memory (hereafter referred to as the ROM), the E register, and the micro-instruction decoding circuits. During operation, macro-instructions obtained from the main memory are used to address the initial instruction of a micro-program residing in the ROM. Sequencing through the micro-program causes micro-instructions to be distributed to the appropriate circuits and perform the actions specified by the particular micro-instructions.

3.3.2 Within the control unit, the Q register is used for temporary storage of a portion of the ROM address, the Z register is used to directly address the ROM, and the B register is used to store the micro-instruction during the decoding and execution.

3.4 Priority and DMA Logic

3.4.1 In a complete system, with buffered I/O controllers etc., many events will be occurring simultaneously and also asynchronously. However, the data bus and the address bus cannot have more than one pair of users at one time. Typically data is transferred to or from main memory while the address is given by the CPU or a Direct Memory Access device. Thus, during each timing cycle only one part of the system is allowed to control the busses. The events that can take place are listed in ascending order of priority.

a) Actual execution of a micro-instruction

b) Fetch a micro-instruction from main memory (micro-instruction mode, see 3.5.1)

c) Halt. The processor will be dormant until the start switch is operated.

d) Direct Memory Access device cycle. A special DMA device is the operator console (optional equipment) through which the contents of any memory location can be displayed and altered. Also, many of the registers in the CPU can be displayed on the console (see product spec. for TPU).

e) Refresh dynamic solid state memory. The CPU will send a five-bit refresh address over the address bus during a refresh cycle.

3.5 Main Memory

3.5.1 The main memory may be from 4K up to 32K eight-bit locations, and is addressed by either the P, the A, or the B register. (Main memory is addressed by the Z register when operating in the micro-instruction mode). The main memory handles the task of storing the program instructions as well as the data during operation, and all information going to and from the main memory passes through the C and D registers. In addition, a micro-instruction mode is provided within the system to enable the main memory to store micro-instructions in a sequence not included in the standard read-only memory. In this mode, the sequence of micro-instructions from main memory control the operation of the system, and the main memory effectively replaces the ROM.

3.6 Data and Program Registers

3.6.1 Figure 1 shows that several registers are included within the processor to permit temporary storage of data, of program instructions, and of memory addresses. Each register, with the exception of the P register, is used for several functions during the execution of the program instructions (macro-instructions).

3.7 Arithmetic - Logic Unit

3.7.1 The ALU is used to perform such operations as addition, subtraction, comparison, and logicals. All data to the ALU passes through the X and Y registers. The arithmetic results appearing at the output are placed on the data bus and onto the register bus for application to the C or D register.

3.8 Data Bus and Controllers

3.8.1 All data and commands between the processor and I/O devices are fed over the data bus which connects the I/O controllers and the C and D registers. The I/O controller provides all interface functions required to operate the processor with a particular I/O device. Each type of device, such as the typewriter, the tape reader/punch, etc. requires a matching controller to provide the proper interface with the processor.

3.8.2 The controllers permit the processor to address and select a particular device, to determine the status of the device, to control the device, and in the case of the typewriter and data set (communications), to translate or condition the data to the required format for transmission in either direction.

4.0 OVERALL OPERATION OF THE PROCESSOR

4.1 Prior to actual operation, the processor is initialized and the desired program is loaded into the designated area of main memory. During initialization, the internal registers are reset to zero and a Read Hex macro-instruction is generated. That is, the operator presses the IPL switch on the processor control panel to generate a general reset to the system; and, also to generate a ROM address that initiates a Read Hex instruction addressed to device zero. The program is then loaded by means of device zero.

NOTE

Refer to the QANTEL V Reference Manual for the description and purpose of each macro-instruction.

4.2 The execution of any program instruction (macro-instruction) is preceded by the fetch cycle. The QANTEL V Processor is a single-address or two address unit, and it is early in the fetch cycle that the processor determines whether the current instruction is single or two-address. (Details of the addressing structure is presented in the QANTEL V Reference Manual.) If the current instruction is single-address (three bytes in length), the fetch cycle loads the three bytes of the instruction into the A, C, D, and L registers (shown in Figure 1) and then initiates execution, using the first 16 positions of main memory (the accumulator) as the implied B operand field. If the current instruction is two-address, the fetch cycle loads the first two bytes of the instruction into the A register, then determines from the most significant four bits of the third byte that the instruction is two-address. Upon detecting the two-address format, the fetch cycle continues fetching to load the A, B, C, D, L, and M registers with the complete two-address instruction (six bytes in length).

NOTE

> In actual operation, only five and one-half bytes of the instruction are placed in registers. The most significant four bits of the third byte are used by the processor only to determine whether or not the instruction is two-address, and having no further purpose, are subsequently destroyed during the fetching of the remainder of the instruction.

4.3 With the fetch cycle completed and the macro-instruction contained in the appropriate registers, execution of the program instruction begins by moving the contents of the C register (op code) through the Q register to the ROM page address portion (Z9 - Z6) of the Z register. The Z register addresses the specified (by the op code) page in the ROM, and the micro-program residing in that page is initiated.

4.4 The ROM is arranged in banks, in pages within the banks, and in words within the pages. (The standard processor contains only one bank of ROM.) The 11-bit Z register is functionally divided into three sections to address the ROM by bank, by page, and by word. The Z10 bit is used to address the required bank of ROM, while the Z9 through Z6 bits are used to address a specific page in the ROM bank. The Z5 through Z0 bits are used to address a specific page word (micro-instruction) within the page.

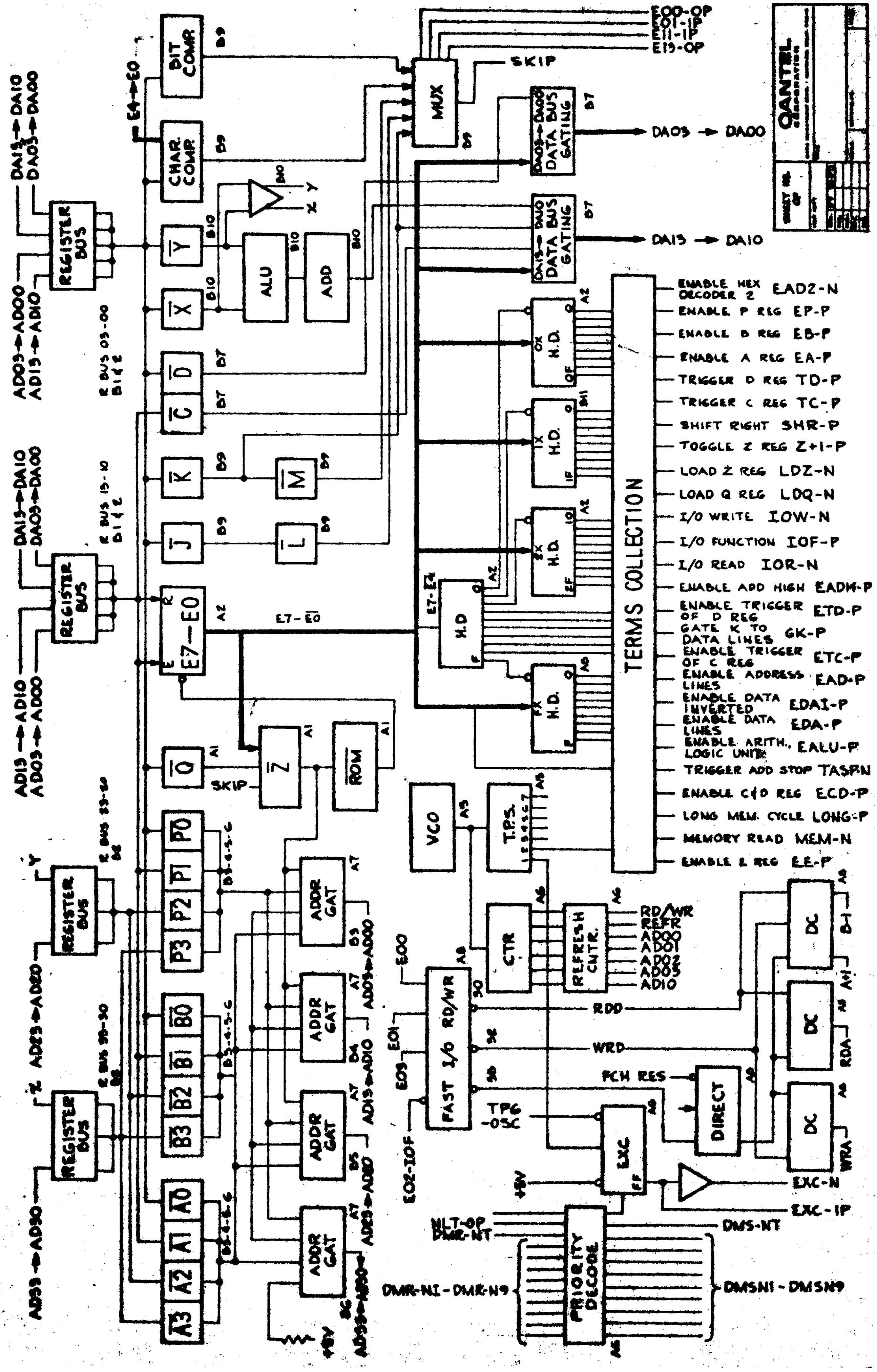
4.5 When the addressed micro-program is initiated, each eight-bit ROM word (micro-instruction) is sequentially addressed and fed to the E register. While held in the E register, the micro-instruction is applied to micro-instruction decoding circuits, where it is decoded to produce an outnut. The decoded output is in turn fed to designated elements of the processor to perform the indicated micro-instruction. When the micro-instruction has been executed, except in the case of a branch micro-instruction or skip function, the Z5 through Z0 portion of the Z register is incremented by one to address the next micro-instruction in the page. The succeeding micro-instruction is then fed to the E register, where it is decoded and executed.

NOTE

> Some micro-instructions test for specific internal conditions, and when true, increment the Z register by two counts to provide a skip function.

4.6 As a result of the incrementing Z register, the micro-program is executed in a step-by-step fashion, operating in conjunction with the previously stored data and instruction information (operand addresses, lengths, op codes, variants, etc.) to perform the specific macro-operation. Upon completion of the micro-program, a "go to fetch cycle" command is brought from the ROM, and the next macro-instruction is brought from main memory as described in paragraph 4.2.

4.7 Each of the macro-instructions supplied with the QANTEL V Processor utilizes a different micro-program, or a variation of some common micro-program residing in the ROM. To fully understand how the processor performs the different macro-instructions, the corresponding algorithms (micro-programs) must be analyzed on a micro-instruction level. At this level of operation, it can be seen how the data is moved within the processor and I/O devices, and how each processor element is used to complete a macro-operation.



QANTEL CORPORATION
SHEET NO. 08
DA15 → DA10
DA05 → DA00
AD05 → AD00
AD15 → AD10
REGISTER BUS
R BUS 05-00 B1 & 2
R BUS 15-10 B1 & 2
R BUS 35-30 B2
R BUS 25-20 B2
AD15 → AD10
AD05 → AD00
AD25 → AD20
AD35 → AD30
Y
Z
E4 → E0
BIT COMP. B9
CHAR. COMP. B9
Y
X
D
C
K
J
B10
B7
B9
ALU
ADD
M
L
X Y
E7 — E0
R
E
A2
E7 - E0
Q
A1
Z
SKIP
ROM
P0
P1
P2
P3
B3-4-5-6
B0
B1
B2
B3
A0
A1
A2
A3
ADDR GAT
A7
B3
AD05 → AD00
AD15 → AD10
AD25 → AD20
AD35 → AD30
B4
B5
B6
+5V
MUX
B9
E00-OP
E01-IP
E11-IP
E15-OP
SKIP
DA05 → DA00 DATA BUS GATING
DA15 → DA10 DATA BUS GATING
DA05 → DA00
DA15 → DA10
0X H.D. OF
1X H.D. 1F
2X H.D. 2F
E7 - E4
H.D.
FX H.D.
A2
A3
TERMS COLLECTION
ENABLE HEX DECODER 2 EAD2-N
ENABLE P REG EP-P
ENABLE B REG EB-P
ENABLE A REG EA-P
TRIGGER D REG TD-P
TRIGGER C REG TC-P
SHIFT RIGHT SHR-P
TOGGLE Z REG Z+1-P
LOAD Z REG LDZ-N
LOAD Q REG LDQ-N
I/O WRITE IOW-N
I/O FUNCTION IOF-P
I/O READ IOR-N
ENABLE ADD HIGH EADH-P
ENABLE TRIGGER OF D REG ETD-P
GATE K TO DATA LINES GK-P
ENABLE TRIGGER OF C REG ETC-P
ENABLE ADDRESS LINES EAD-P
ENABLE DATA INVERTED EDAI-P
ENABLE DATA LINES EDA-P
ENABLE ARITH. LOGIC UNIT EALU-P
TRIGGER ADD STOP TASP-N
ENABLE C/D REG ECD-P
LONG MEM. CYCLE LONG-P
MEMORY READ MEM-N
ENABLE E REG EE-P
VCO
A5
T.P.S. 1 2 3 4 5 6 7
A6
CTR
REFRESH CNTR.
RD/WR
REFR
AD00
AD01
AD02
AD03
AD10
E00
E01
E05
E02-IOF
FAST I/O RD/WR
A8
RDD
WRD
FCH RES
DIRECT
TP6 -OSC
EXC
EF
+5V
EXC-N
EXC-IP
DC
D-I
AI
RDA
WRA
NLT-OP
DMR-NT
DMS-NT
PRIORITY DECODE
DMR-N1 - DMR-N9
DMSN1 - DMSN9
A6

Table 1 Micro-Instruction Set.

MNEMONIC	OP CODE (HEXADECIMAL)	FUNCTION
BRC	00	Branch to the ROM page specified by the C register
CK	01	Move the contents of the K register to the C register if SW1 = 1; or to the D register if SW1 = 0.
BIN	02	Place the binary result from the arithmetic unit (AU) in the C or D register, use SW1.
AND	03	Place the AND result from the AU in the C or D register, use SW1.
OR	04	Place the OR result from the AU in the C or D register, use SW1.
DEC	05	Place the decimal result from the AU in the C or D register, use SW1.
XOR	06	Place the exclusive OR result from the AU in the C or D register, use SW1.
HLT	07	Halt the processor on the next branch to fetch.
GAL	08	Move A1 and A0 to the C and D registers.
GAH	09	Move A3 and A2 to the C and D registers.
GBL	0A	Move B1 and B0 to the C and D registers.
GBH	0B	Move B3 and B2 to the C and D registers.
GPL	0C	Move P1 and P0 to the C and D registers.
GPH	0D	Move P3 and P2 to the C and D registers.

NOTE

The make up of the A, B, and P registers is shown in the block diagram, Figure 4-1.

GC	0E	Move the contents of the C register to the D register.
FCH	0F	Branch to instruction fetching.
L-1	10	Subtract one from the L register contents.
L+1	11	Add one to the L register contents.
K-1	12	Subtract one from the K register contents.
K+1	13	Add one to the K register contents.
M-1	14	Subtract one from the M register contents.

MNEMONIC	OP CODE (HEXADECIMAL)	FUNCTION
M+1	15	Add one to the M register contents.
TSK	16	Skip the next micro-instruction if the K register contents are zero.
TSL	17	Skip if the L register contents are zero.
TSM	18	Skip if the M register contents are zero.
SHR	19	Shift the contents of the C and D registers one bit to the right; the high order bit of C is set to 1 if switch five is on.
LDK	1A	Move the contents of the D register to the K register.
LDM	1B	Move the contents of the K register to the M register.
LDX	1C	Move the contents of the D register to the X register.
LDY	1D	Move the contents of the D register to the Y register.
LDJ	1E	Move the contents of the D register to the J register.
LDL	1F	Move the contents of the J register to the L register.
TS1	20	Skip if SW1 is off.
TS2	21	Skip if SW2 is off.
TS3	22	Skip if SW3 is off.
TS4	23	Skip if SW4 is off.
TS5	24	Skip if SW5 is off.
TS6	25	Skip if SW6 is off.
TS7	26	Skip is SW7 is off.
TR1	27	Trigger SW1 to the opposite state.
TR2	28	Trigger SW2 to the opposite state.
TR3	29	Trigger SW3 to the opposite state.
TR4	2A	Trigger SW4 to the opposite state.
TR5	2B	Trigger SW5 to the opposite state.
TR6	2C	Trigger SW6 to the opposite state.
TR7	2D	Trigger SW7 to the opposite state.
RSW	2E	Reset SW1 through SW6 to off (SW7 is reset by FCH).
BIX	2F	Compare the contents of the X and Y registers, skip if no 1 bit matching.
RDD	30	Read data from I/O bus into the C and D registers, send strobe.
RSO	31	Read status of I/O into the C and D register, send strobe.

MNEMONIC	OP CODE (HEXADECIMAL)	FUNCTION
WRD	32	Move data from the C and D registers onto the I/O bus, send strobe.
WRC	33	Move command from the C and D register onto the I/O bus, send strobe.
RIO	34	Reset I/O device.
SRD	35	Send read command to the I/O device, send strobe.
SWR	36	Send write command to the I/O device, send strobe.
TER	37	Send terminate command to the I/O device, send strobe.
DIR	38	Initiate direct read or write.
RS1	39	Read I/O status 1 from the I/O bus into the C and D registers, send strobe.
SEL	3F	Select the I/O device specified by the D register.
BRO	4X	Branch to the first quarter of a page.
BR1	5X	Branch to the second quarter of a page.
BR2	6X	Branch to the third quarter of a page.
BR3	7X	Branch to the fourth quarter of a page.
BIC	8X	Compare the contents of the C register and the E00 through E03 bits of the E register; skip if no 1 bit matching.
BID	9X	Same as BIC but with the D register instead of the C.
HXC	AX	Compare the contents of the C register and the E00 through E03 bits of the E register; skip if all bits not matching.
HXD	BX	Same as HXC but with the D register instead of the C.
NPG	CX	Move a page address from the E00 through E03 portion of the E register to the Q register.
LDC	DX	Move a constant to the C register, take E00 through E03.
LDD	EX	Move a constant to the D register, take E00 through E03.
MIM	F0	Initiate micro-instruction mode, (take next micro-instruction from location 2048 of main memory).
RDA	F1	Read from main memory using address in the A register, data to the C and D registers.
RDB	F2	Read from main memory using address in the B register, data to the C and D registers.

MNEMONIC	OP CODE (HEXADECIMAL)	FUNCTION
RDP	F3	Read from main memory using address in the P register, data to the C and D registers. Increment P
WRA	F4	Write to main memory using address in the A register, data from the C and D registers.
WRB	F5	Write to main memory using address in the B register, data from the C and D registers.
LDP	F6	Move the contents of the A register to the P register.
LDB	F7	Move the contents of the A register to the B register.
LDA	F8	Move the contents of the X, Y, C, and D registers to the A register (A3, A2, A1, and A0).
A+1	F9	Add one to the A register contents.
A-1	FA	Subtract one from the A register contents.
B+1	FB	Add one to the B register contents.
B-1	FC	Subtract one from the B register contents.
NBK	FD	Switch to other ROM bank on the next BR micro-instruction.
WRP	FE	Write to main memory using the address in the P register, data from the C and D registers.
P-1	FF	Subtract one from the P register contents.

DETAIL SPECIFICATION

The CPU consists of two 12" X 14" printed circuit boards, CPU A and CPU B, that plug into the processor housing. The housing, 26" X 15" X 17", which may be mounted in a standard "L" shaped secretarial desk, also contains from 4K to 32K of memory, the optional Operator Panel interface (TPU) and up to twelve I/O device controllers.

Word Length: 8 bits

Internal control by read-only memory and micro-instructions

Micro Instructions: 86

Macro Instructions (ROM): 46

Interrupt feature included

Direct Memory Access channels: 9 max (Each will replace one normal I/O).

DMA Service Time: 1.5 µs/word min.

I/O Service Time: 7 µs/word (read or write normal)

I/O Service Time: 19 µs/word (read hex)

I/O Service Time: 26 µs/word (write hex)

Memory Cycle Time: 1.5 µs

Execution Time: 1.0 µs