NORD-100
Microprogramming
Description

NORSK DATA A.S

NORD-100
Microprogramming
Description

NOTICE

The information in this document is subject to change without notice. Norsk Data A.S assumes no responsibility for any errors that may appear in this document. Norsk Data A.S assumes no responsibility for the use or reliability of its software on equipment that is not furnished or supported by Norsk Data A.S.

PRINTING RECORD

Printing	Notes
01/80	Original Printing

NORD-100 MICROPROGRAMMING DESCRIPTION
Publication No. ND-06.018.01

NORSK DATA A.S
P.O. Box 4, Lindeberg gård
Oslo 10, Norway

Manuals can be updated in two ways, new versions and revisions. New versions consist of a complete new manual which replaces the old manual. New versions incorporate all revisions since the previous version. Revisions consist of one or more single pages to be merged into the manual by the user, each revised page being listed on the new printing record sent out with the revision. The old printing record should be replaced by the new one.

New versions and revisions are announced in the ND Bulletin and can be ordered as described below.

The reader's comments form at the back of this manual can be used both to report errors in the manual and to give an evaluation of the manual. Both detailed and general comments are welcome.

These forms, together with all types of inquiry and requests for documentation should be sent to the local ND office or (in Norway) to:

Documentation Department
Norsk Data A.S
P.O. Box 4, Lindeberg gård
Oslo 10

TO THE READER

The NORD-100 Microprogramming Description manual describes how to micro-program the NORD-100 in order to take full advantage of the writable control store feature. This manual should be of interest to all technical personnel writing their own microprogram. It will also be useful for those who want a more detailed understanding of the microprogrammed structure of the NORD-100 CPU.

PREREQUISITE KNOWLEDGE

A general knowledge of the NORD-100 computer system, together with some basic understanding of the NORD-100 CPU and some experience in digital techniques is recommended.

THE MANUAL

Chapter 1 is a short introduction to the writable control store option, while Chapter 2 gives an introduction to the CPU architecture. The microprogram word format is described in Chapter 3 and how to write the microprogram is found in Chapter 4. Chapter 5 shows the usage of the writable control store.

In the appendixes are some data sheets for the main building blocks in the CPU, together with the microprogram flow charts. The data sheets are taken from "AM2900 Bipolar Microprocessor Family" issued by Advanced Micro Devices, Inc. More information about the circuits will be found in this publication. Data sheets for the microprogram sequencer are found in "The TTL Data Book" issued by Texas Instruments.

TABLE OF CONTENTS

+ + +

Section:		*Page:*
1	INTRODUCTION	1—1
2	CPU ARCHITECTURE INTRODUCTION	2—1
3	MICROPROGRAM WORD FORMAT	3—1
4	WRITING THE MICROPROGRAM	4—1
5	WRITABLE CONTROL STORE USAGE	5—1
Appendix:		
A	MNEMONIC LIST	A—1
B	DATA FOR AM 2901 BIT SLICE	B—1
C	DATA FOR AM 2914 INTERRUPT CONTROLLER	C—1
D	FUNCTIONAL BLOCK DIAGRAM FOR 74S482 SEQUENCER	D—1
E	MICROPROGRAM FLOW CHARTS	E—1
F	MAIN MEMORY/WCS CORRESPONDENCE	F—1

1 INTRODUCTION

The NORD-100 CPU with the Writable Control Store option, has its basic microprogram contained in the lower 2K of the microprogram addressing area (addresses 0 - 3777). The Writable Control Store (WCS) contains the addresses 4000 - 4377.

In order to write micro code for the WCS, the following steps must be undertaken:

1. Specification of the new machine instruction that will be inplemented by the new microprogram.

2. Design and coding of the necessary microinstructions to perform the machine instructions.

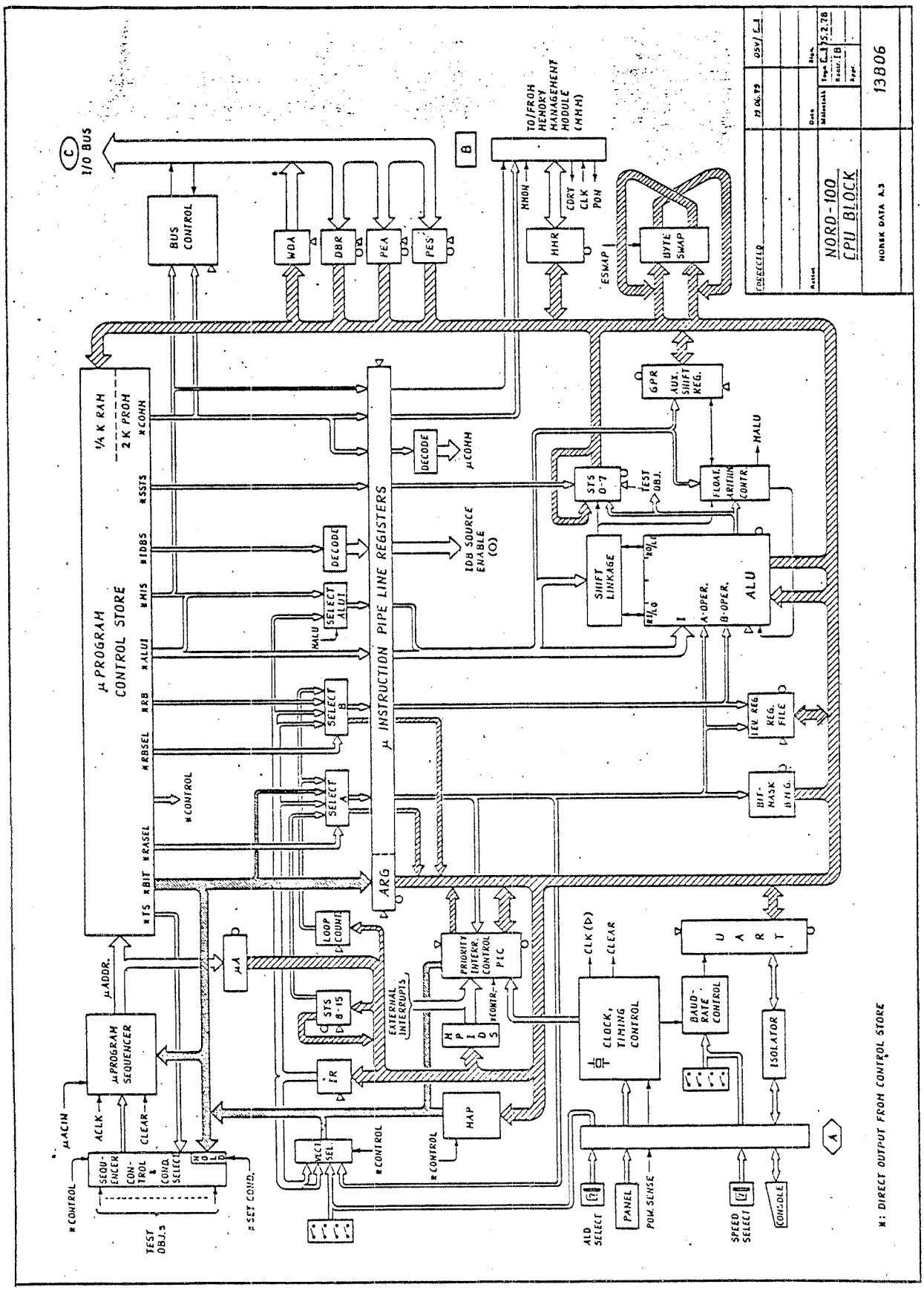
3. Assembly of the symbolic microprogram into binary format with the NORD-100 WCS assembler.

4. Place the binary representation of the WCS contents in main memory and load it into WCS with the load writable control store machine instruction.

5. Use the Customer Specified Instructions to execute the microprogram now residing in the WCS.

2 CPU ARCHITECTURE INTRODUCTION

The CPU consists of elements built around the Internal Data Bus, IDB. IDB is shown hatched in Figure 2.1. A few comments to each of the elements in Figure 2.1 follows.


NORD-100
CPU BLOCK
NORSK DATA A.S
13806
I/O BUS
TO/FROM MEMORY MANAGEMENT MODULE (MMM)
BUS CONTROL
WDA
DBR
PEA
PES
MMON
MMR
CONT
CLK
PON
BYTE SWAP
ESWAP
GPR
AUX. SHIFT REG.
DECODE
μCOMM
STS 0-7
TEST OBJ.
FLOAT. ARITHM CONTR.
MALU
IDB SOURCE ENABLE (O)
SHIFT LINKAGE
A-OPER.
B-OPER.
ALU
SELECT ALU
SELECT B
SELECT A
IEN REG
REG. FILE
BIT-MASK
μINSTRUCTION PIPE LINE REGISTERS
ARG
μPROGRAM CONTROL STORE
1/4 K RAM
2 K PROM
μADDR.
μPROGRAM SEQUENCER
μA
LOOP COUNT
PRIORITY INTERR. CONTROL PIC
EXTERNAL INTERRUPTS
HPIDS
STS 8-15
IR
MAP
CLOCK, TIMING CONTROL
BAUD-RATE CONTROL
UART
ISOLATOR
CLK (P)
CLEAR
μACIN
ACLK
CLEAR
SEQUENCER CONTROL & COND. SELECT
TEST OBJ.S
SET COND.
ALD SELECT
PANEL
POW. SENSE
SPEED SELECT
CONSOLE
*: DIRECT OUTPUT FROM CONTROL STORE

ALU performs all arithmetic and logic functions. Two 4 bit operand pointers, the A and B operands, select sources for operations to be performed. Nine ALU function bits select operations within the ALU. The data input and output ports are directly connected to IDB.

SHIFT LINKAGE controls the shift mode as well as the length of the shift operands.

STS 0 - 7 contains the lower byte of the STS register.

FLOATING ARITHMETIC CONTROL performs the necessary functions to speed multiply and divide operations.

GPR is a General Purpose Register used by the microprogram for various purposes. It will always be shifted the same way as an ALU register if shift is specified in the ALU control. GPR is especially valuable during multiply and divide operations. In multiply operations it holds one of the operands and in divide operands it gets the result shifted in from the right.

MMR is the communicating register against the memory management module.

BUS CONTROL controls the access to the system bus, which links the CPU and peripheral interfaces, memory modules. etc.

PES and *PEA* keeps information about system bus abnormalities.

WDA and *DBR* keeps data going between IDB and the system bus.

μPROGRAM CONTROL STORE holds the microprogram controlling the CPU.

μPROGRAM SEQUENCER generates the address of the next microinstruction to be executed. This sequencer can get the address from the microprogram (jump), from the current address + 1 (continue) or from an internal stack, which is 4 deep, and can be pushed or popped.

SEQUENCER CONTROL and *CONDITION SELECT* controls conditional branching in the microprogram sequence. A code indicating a test object is outputted to these circuits from the microprogram, together with a code indicating what to do if a later test gives a false result. When a test is enabled at a later state, the result of the former ALU operation is tested and may result in conditional branching.

VECTOR SELECT controls the source of the 4 lower address bits in a vectorized jump sequencing microinstruction. The 8 uppermost address bits are specified in the microinstruction (in the microinstruction the 4 lower bits must be 0). In this manner, a 16 way branch is effectively implemented.

MAP contains the entry points of all machine instructions. It is used to generate entry point addresses whenever a machine instruction is to be executed.

PIC contains the interrupt system from level 10 and upwards.

MPIDS can set interrupt bits in the PIC.

IR contains the 10 least significant bits of a machine instruction.

STS 8 - 15 contains the upper byte in the STS register.

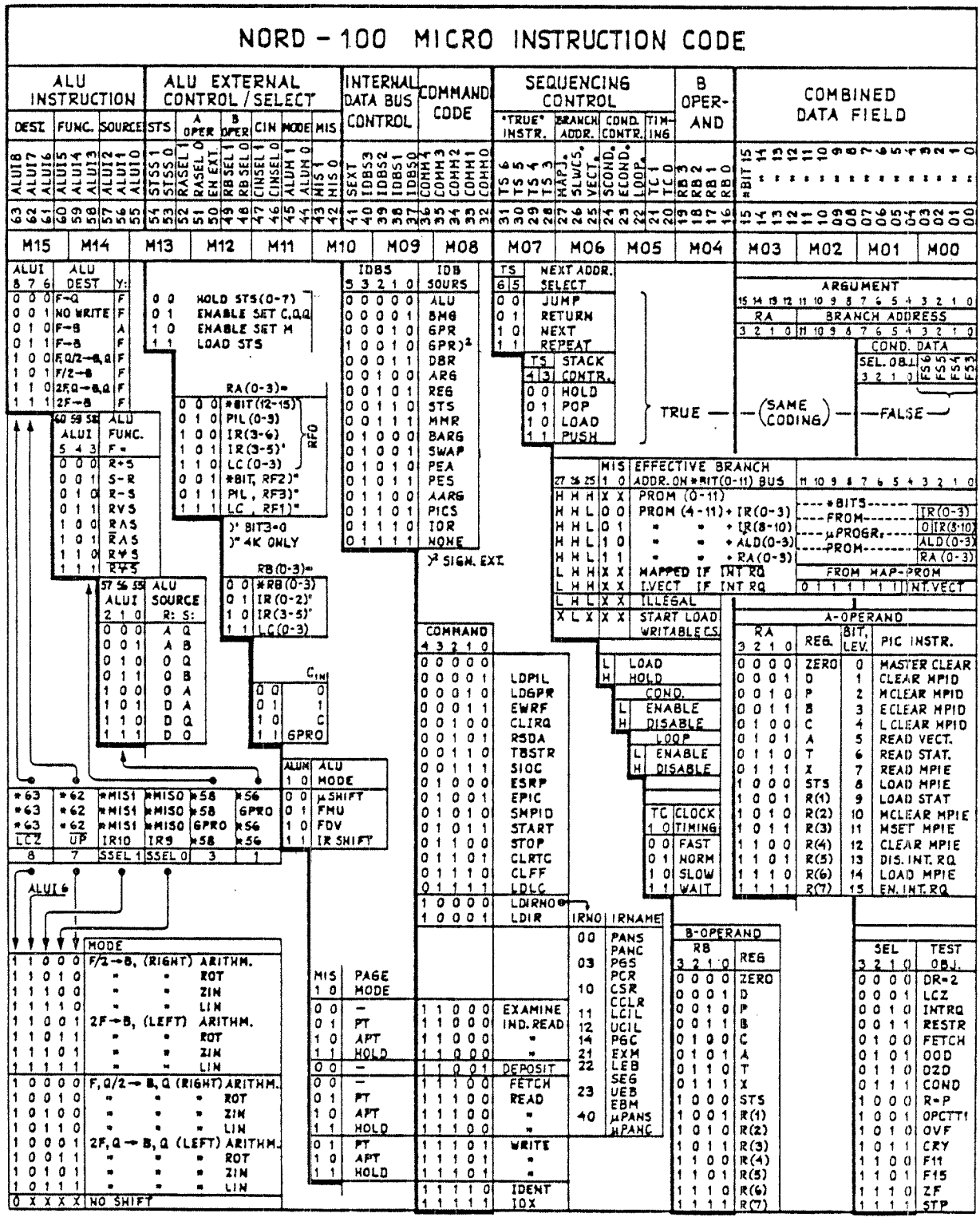
LOOP COUNTER is a 6 bit register with the upper bit as sign bit. It has an auto increment/decrement facility which makes it count towards zero when counting. Zero inhibits further counting. The loop counter is used to repeat one (or a few) microinstructions a predetermined number of times, or to count, for example, the number of shifts needed to normalize a floating number.

BMG (Bit Mask Generator) generates a 1 bit among zeros. The position of the 1 bit is controlled by the A operand.

REG. FILE contains the register blocks of the interrupt levels not presently active. It also contains scratch regsiters for MOPC and for the interrupt system. It is addressed using the A oprerand to indicate the level, and the B operand to indicate the register number within that level.

3 MICROPROGRAM WORD FORMAT

Each microprogram word is 64 bits wide. The width is divided into fields controlling different functions within the CPU. These fields and their functions are described in this chapter. The description frequently refers to Figure 3.1 which is a drawing of a microprogram word.



Bits 55 - 63, together with bits 44 - 45, controls the behaviour of the ALU. The ALU is built of the bit slice component 2901, and the functional description of 2901 is included as Appendix B.

The effects of bits 55 - 63 are explained in Appendix B, with some additional information in the following pages.

Bits 44 and 45 modifies bits 55 - 63 before they reach the ALU.

ALUM0 = 0 (bit 44)
ALUM1 = 0 (bit 45): Micro Controlled Mode

The ALU behaviour is controlled entirely from the microprogram.

ALUM0 = 1 (bit 44)
ALUM1 = 0 (bit 45): Multiply Mode

The ALU behaviour is controlled from the microprogram, except that GPR bit 0 controls the addition of the A operand register or 0. This is only used in multiplication instructions.

ALUM0 = 0 (bit 44)
ALUM1 = 1 (bit 45): Divide Mode

The ALU behaviour is controlled from the microprogram except that GPR bit 0 controls whether or not the A operand register should be added or subtracted. This is only used in divide instructions.

ALUM0 = 1 (bit 44)
ALUM1 = 1 (bit 45): Shift Instruction (machine instruction is shift)

The shift direction and shift mode is controlled from the loop counter and the bits 9 - 10 of the machine instruction. The extra microinstruction executed after the loop counter reaches 0 will not shift the ALU result.

Bits 53 - 54 controls the behaviour of the 8 lower bits of the STS register.

00	will not affect the STS register
01	will update the C, O and Q flip-flops to the result of the current ALU instruction
10	will update the M flip-flop to the disappearing bit in a shift micro-instruction. If there is ALUM IRSHIFT, M will be updated regardless of the setting of bits 53 - 54.
11	will load the 8 lower bits of the STS register from IDB

Bits 50 - 52 controls the source of the A operand to the ALU.

000 The A operand is bits 12 - 15 of the microinstruction.

010 The A operand is the PIL register (used to address the current level register block in the register file).

100 The A operand is bits 3 - 6 of the current machine instruction. Used in bit operation instructions and in instructions specifying a level in the register file.

101 The A operand is bits 3 - 5 of the current machine instruction. This is used to address the source register in machine instructions.

110 The A operand is bits 0 - 3 of the loop counter.

Bits 48 - 49 controls the source of the B operand to the CPU.

00 The B operand is bits 16 - 19 of the microinstruction.

01 The B operand is bits 0 - 2 of the current machine instruction. Used to address the destination register in machine instructions.

10 The B operand is bits 3 - 5 of the current machine instruction. Used to address the source register in machine instructions.

11 The B operand is bits 0 - 3 of the loop counter. This is used to read the loop counter by using BARG as IDB source.

Bits 46 - 47 control the carry in line to the ALU.

00 Carry is 0
01 Carry is 1
10 Carry is the C flip-flop of the STS register
11 Carry is the GPR bit 0 (used in division)

Bits 42 - 43 control the shift type in shift microinstructions, if the machine instruction is not a shift instruction. They indicate which page table to use together with memory requests.

00 Arithmetic shift/Fetch memory request
01 Rotational shift/Request through PT
10 Zero-end-input shift/Request through APT
11 Link-end-input shift/Request through the page table last specified

Bits 37 - 41 control which source should control the content of IDB (refer to Figure 2.1).

0	ALU	enables its result onto IDB
1	BMG	a 1 bit among zeros is enabled onto IDB. The position of the 1 bit is controlled by the A operand.
2	GPR	the general purpose register is enabled onto IDB.
22	GPR	with bits 8 - 15 equal to bit 7 is enabled onto IDB. This makes sign extension easy during address calculations and argument instructions, etc.
3	DBR	the system bus data read register is enabled onto IDB. This must be used with the WAIT for data ready timing specification.
4	ARG	The 16 lower bits of the microinstruction is enabled onto IDB. This makes it easy to generate 16 bit arguments within the microprogram.
5	REG	The register file address selected by the A and B operands is enabled onto IDB.
6	STS	The 16 bit STS register is enabled onto IDB.
7	MMR	The memory managment read register is enabled onto IDB.
10	BARG	The B operand is regarded as an argument and enabled onto IDB. In this way arguments between 0 - 17 may be generated.
11	SWAP	A byte swap circuit is enabled onto IDB. The two bytes present on IDB during the last cycle have exchanged positions.
12	PEA	The inverse of the PEA register is enabled onto IDB.
13	PES	The inverse of the PES register is enabled onto IDB.
14	AARG	The A operand is regarded as an argument shifted 3 bits left, and enabled onto IDB. In this way arguments between 0 - 170 with 0 in the 3 lower bits may be generated.
15	PICS	The internal status of the PIC is enabled onto IDB. The RSTS command must be issued to the PIC simultaneously.
16	IOR	The register containing UART data and status is enabled onto IDB.
17	NONE	IDB is disabled. Read commands to the PIC may cause PIC to enable internal PIC registers onto IDB.

Bits 32 - 36 specifies a command to be performed by the microinstructions. The commands are:

0	NONE	No command executed.
1	LDPIL	The upper byte of the STS register is loaded with the upper byte of IDB. The new PIL will not be available as A operand before in the microinstruction following the next microinstruction.
2	LDGPR	IDB is loaded into GPR. If a shift is specified in the same microinstruction, GPR will not be shifted, only loaded.
3	EWRF	IDB is written into the register file word specified by the A and B operands.
4	CLIRQ	Inhibits the ability of interrupts to modify the microinstruction sequence at the time of a mapped jump. Used to avoid interrupts when the MAPJ bit is used at other times than FETCH. CLIRQ will also block PANEL interrupts from being included in the IRQ condition.
5	RSDA	Reset the data available signal from the UART on the CPU board.
6	TBSTR	Transmit Buffer Strobe. Transfers the lower IDB byte to the UART output part.
7	SIOC	Set I/O control. Transfers IDB to control several functions. The bits of IDB have these functions. Bit 0: PIN (prepare interrupt) signal for clock. Bit 1: PIN signal for terminal input. Bit 2: PIN signal for terminal output. Bit 3: RFT (ready for transfer) signal for clock. Bit 4: Green LED on CPU (OK). Bit 5: Red LED on CPU (ERROR). Gives master clear on system bus. Bit 6: When = 1, inhibits MOPC input when in continue. Bit 7: Resets the counters giving 50 Hz clock frequency. Used by IOX 11. Bit 10: Not used. Bit 11-12: 0 0 5 bits UART character length 0 1 6 bits UART character length 1 0 7 bits UART character length 1 1 8 bits UART character length Bit 13: When 0 — 1 stop bit; When 1 — 2 (1.5 at 5 bits) stop bits. Bit 14: An extra bit is added to give even parity. The latest IDB value outputted by a SIOC command will always be saved in a word named STATUS in the register file.
10	ESRP	A special flip-flop, the R = P flip-flop, is set to 1 if the ALU result is 0. Used to detect that the next location (where prefetch has already been performed) has been written into (STA * + 1).

11	EPIC	indicates that A operand should be a command to the PIC. The specifications of the PIC are included in Appendix C. If the PIC command is a read command, other than read status, the PIC will enable the register onto IDB regardless of the microprogram bits 37 - 41.
12	SMPID	sets bits in the micro PID register in the PIC. Bits can only be set to 1 by this command. (The PIC command MCLPID can reset bits to 0.)
13	START	resets the STOP flip-flop. The CPU will start executing a main memory program.
14	STOP	sets the STOP flip-flop. Whenever a FETCH has been performed, the microprogram is forced to microaddress 3760 or 3770 by the interrupt hardware. The stop routine is executed.
15	CLRTI	resets the clock flip-flop. Every 20 ms the microprogram is forced to microaddress 3761 or 3771, thereby entering the clock routine. This routine sends out the CLRTI command.
16	CLFF	clears flip-flops having special functions regarding the floating point rounding indicator (TG). The DZD (double zero detect used in FDV) and the OOD (one out detect used in FAD, FSB and FMU) flip-flops are cleared by CLFF.
17	LDLC	loads the loop counter with the 6 lower bits of IDB. The modified loop counter will not be available as A operand before in the microinstruction following the next microinstruction.
20	LDIRNO	select register on memory management module to be accessed later. IDB is transferred to the selector on MMM.
21	LDIR	(or XMM in symbolic language) will transfer IDB to the MMM register selected by a previous LDIRNO command. (To read a selected MMM register, only the IDB source designation MMR should be used.)
30	EXRQ or IRDRQ	differ according to the contents of bits 42 - 43: 00 EXRQ, examine memory request. The physical address must be on IDB. The segment number must already be loaded into the MMM segment register. 01 IRDRQ, PT, indirect address read request with mapping through the normal page table. 10 IRDRQ, APT, indirect address read request with mapping through the alternative page table. 11 IRDRQ, HOLD, indirect address read request with mapping through the last used page table.

31	DERQ	deposit memory request. The physical address must be on IDB, in the segment already loaded into the MMM segment register.
34	RDRQ	mapped read request has four variants, depending on bits 42 - 43: 00 Fetch request 01 Read request mapped through the normal page table. 10 Read request mapped through the alternative page table. 11 Read request mapped through the last used page table.
35	WRRQ	mapped write request has 3 variants, depending on bits 42 - 43: 01 Write request mapped through the normal page table 10 Write request mapped through the alternative page table 11 Write request mapped through the last used page table
36	IDENT	IDENT bus request. As with write requests, the write data must be enabled onto IDB in the next microinstruction, accompanied by a WAIT cycle. The following instruction must also be a WAIT cycle, with DBR as source to IDB.
37	IOX	IOX bus request. Data out and in with be present during the next 2 microinstrutions. The timing, as seen from the microprogram, will be identical with IDENT.

Bits 28 - 31 control the microprogram sequencer. The sequencer is documented in Appendix D. Bits 30 - 31 control the next instruction selection and bits 28 - 29 control the internal stack. These bits will only be in control as long as bits 22 - 23 are high.

Bit 27 is active low, and will load IR from IDB in the next instruction. It will clear the R = P flip-flop. It sets the false sequence control to JUMP and POP, and the text object to DR = 2 (used by ROP and BOP instructions to enabled test of the P register as destination). If the next microinstruction is a JMP, the address will be taken from the mapping PROM, or be generated by the interrupt hardware if CLIRQ command has not been given.

Bit 26 is active low, and will start the loading of the WCS from main memory. The next microinstruction must be a jump to microaddress 7400. Thereafter, the CPU hardware goes into a special state and moves the memory addresses 36000 - 37777 into WCS. The microprogram continues from microaddresses 0 when the loading of WCS is finished.

Bit 25 is active low and specifies vectorized jump. The 8 upper bits of the jump microaddress are normally taken from micro word bits 4 - 11. The lower 4 bits can be selected from 4 sources, controlled by bits 42 - 43.

00 IR 0 - 3 controls the vector, used in TRA and TRR.
01 IR 8 - 10 controls the vector, used for address calculation in memory reference instructions.
10 ALD 0 - 3, the ALD setting controls the vector.
11 The A operand bits control the vector.

Bit 24 is active low and sets condition to be tested on a later occasion. A 4 bit code indicating test condition and a 4 bit code indicating the sequence instructions in case of a false test, are outputted to a hold register.

Bit 23 is active low and enables testing (against the previously set test condition) of the result of the micro instruction before the micro instruction containing this test. If the test is true, the sequencer executes the bits 28 - 31 to find the next microinstruction. If the test is false, the sequencer executes the false sequence instruction previously outputted to the hold register by bit 24.

Bit 22 is active low and increments or decrements the loop counter (always towards 0). If the selected condition is false, RETURN, HOLD is executed by the sequencer. If it is true, bits 28 - 31 are executed. The new value of the loop counter will not be available as A operand or test object before in the microinstruction following the next microinstruction.

Bits 20 - 21 controls the duration of each microinstruction. It also controls the synchronization between the CPU and the system bus control.

00 specifies fast instruction, approx. 150 ns
01 specifies intermediate instruction, aprox. 170 ns
10 specifies slow instruction, approx. 190 ns
11 specifies wait for bus control synchronization

After a read memory request, one WAIT cycle must occur to finish the bus request. If the result is needed in the CPU, WAIT must be accompanied by DBR as IDB source.

After a write memory request, the data to write must be enabled onto IDB in the next microinstruction, accompanied with a WAIT cycle. Thereafter, another WAIT cycle must be encountered before a new request is sent out.

As long as the above rules are obeyed, WAIT cycles may be used freely, with no effect.

In NORD-100 all microinstructions are either "slow" or "wait".

In NORD-100, with fast option, most microinstructions are "fast", while the same are "wait" as in the ordinary NORD-100. A few microinstructions have to be "slow" because of component timing characteristics. This applies to:

1. All microinstructions handling the UART (SIOC, RSDA, TBSTR).

2. Microinstructions where the IDB source is used by the ALU as one of the operands in a arithmetic operation, and the result will be tested by the F = 0 condition in the next microinstruction.

3. All microinstructions that involve data exchange with the memory management module (microinstructions containing IDBS, MMR or COMM, LDIR or COMM, XMM).

Bits 16 - 19 control the B operand as long as bits 48 - 49 are zero.

Bits 12 - 15 control the A operand as long as bits 50 - 52 are zero. The A operand controls BMG and acts as instruction to the PIC (see Appendix C).

Bits 0 - 11 contains the jump address in jump microinstructions. The jump addresses may also be generated by the mapping PROM or the interrupt hardware, and partly by the vectoized jump mechanism. When these sources generate jump addresses, the corresponding microword bits must be 0.

Bits 0 - 3 selects the false sequence to be outputted by bit 24 to the hold mechanism. It specifies the sequencer instruction in case a later test (bit 23) gives a false result.

Bits 4 - 7 selects the test object to be outputted by bit 24 to the hold mechanism. If a test object changes state, one microinstruction will be executed before the change is detected. The cause of this is the pipelining of ALU and sequencing calculations.

The 16 test objects are:

0	DR = 2	destination register is P. Used to discard prefetch if P is destination in ROP and BOP.
1	LCZ	loop counter is 0. Used to go through a loop a predetermined number of times.
2	IRQ	used to test the interrupt request flip-flop, to see if interrupts are pending. If the test is preceded by the CLIRQ command, only interrupts from AM 2914 are tested and not panel interrupts such as STOP, RTC, etc.
3	RESTR	used to test if the current program is running in restricted mode. Instructions that are privileged should not execute when this condition is true. Privileged WCS instructions must test this condition, other privileged instructions are mapped different whether RESTR is on or off.
4	FETCH	indicates whether the last memory access was a fetch of new instruction. Used to generate correct P in case of page fault.
5	OOD	the one out detect flip-flop is used to set TG correctly in FAD, FSB and FMU instructions.
6	DZD	the double zero detect flip-flop is used to set TG correctly in FDV instructions.
7	COND	This condition will latch the true/false result of the last microinstruction so that it can be tested later if that is more convenient than testing it now. It will postpone the test as long as desired.
8	R = P or ZF	Used to detect if single or multiple store instruction has affected * + 1. If true, the prefetched instruction must be discarded.

9	OPCTT1	is used to test whether device number 300 - 307 should use the MOPC terminal. If a strap on the CPU card is removed, this line will never be true. If the strap is intact, the line will be true when device numbers 300 - 307 are encountered, indicating that this device is not on the system bus but directly micro-program controlled.
10	OVF	overflow from ALU. Indicates overflow during arithmetic operations.
11	CRY	carry out from ALU.
12	F11	bit 11 from ALU.
13	F15	bit 15 (sign bit) from ALU
14	ZF	Result = 0 from ALU.
15	STP	indicates that the STOP flip-flop is 1.

Bits 0 - 15 will be enabled onto IDB as an argument if ARG is IDB source.

4 WRITING THE MICROPROGRAM

Assembly Language Considerations

At this point, it is necessary to go carefully through the mnemonic list given in Appendic A and compare the octal values corresponding to the mnemonics with Figure 2.1. This gives an understanding of which mnemonic performs what function in the microprogram. It is important to be aware that bits 22 - 27 are inverted after the word is assembled.

All the mnemonics in the list are translated by the NORD-100 WCS assembler to its corresponding octal value. All the octal values of the mnemonics in one microinstruction are "OR"ed together. If an octal number is encountered, it is placed in bits 0 - 15 and regarded as an argument. If a symbolic name is encountered which is not found in the mnemonic list, it is regarded as a label, which is or will be defined. When there is no more input to the WCS assembler, all labels are replaced by their values, which are filled into bits 0 - 11 in the correct microaddresses. The characters accepted by the NORD-100 WCS assembler are "space" and "tab". These are used as delimiters between mnemonics, labels and numbers. "%" is used to indicate that the rest of the line is a comment. Any other characters are used to form character groups, which may be mnemonics, labels and commands. If only numbers between 0 - 7 are found in a character group, it is an octal number.

If "/" is found in a character group, the current location counter (CLC) is updated to the octal number in front of /.

If ":" is found in a character group, the rest of the character group is regarded as a label.

The character ";" terminates each microinstruction and CLC is then incremented. Each microinstruction may contain an unspecified number of lines.

Microprogram Architecture Considerations

Whenever the microprogram jumps to execute one machine instruction, the jump address is taken from the mapping PROM. The addresses for the customer specified instructions are found in WCS in the addresses 4001 to 4015 (refer to the NORD-100 Reference manual). After WCS has been loaded, execution of one of the customer specified instructions will cause the microprogram to branch to one of the microaddresses 4001 - 4015. When this point is entered, some of the CPU scratch registers have defined value:

1. GPR contains the machine instruction.
2. False sequence is JMP and POP.
3. R = P flip-flop is cleared.
4. Test object is DR = 2.
5. IR contains the lower 11 bits of the machine instruction.
6. A FETCH request is sent out for the next machine instruction.
7. The P register points to the next machine instruction.
8. The top word of the sequencer stack contains the microaddress PMODX.
9. The next to the top word of the sequencer stack contains the microaddress USDBR.

When the microinstructions in WCS are finished doing their task, return to the PROM resident part of the microprogram should take place in an orderly manner. There are five natural return points:

USDBR:

The system bus must not have been used by the WCS instructions. No WAIT cycle or DBR reading must have been done. The fetch request sent along before the jump to WCS will give the next machine instruction. The sequence stack must be popped once, to bring USDBR to the top word.

USGPR:

The WCS instructions must have read the DBR onto IDB, with an accompanying WAIT cycle, and saved it in GPR. Thereafter the WCS routines may have used the system bus for other purposes. The next machine instruction to execute will be the one requested before the WCS instruction was entered. The sequencer stack must be popped once to bring USDBR to the top word.

INIT:

The WCS instructions must have had a WAIT cycle to end the fetch bus cycle that was set up before WCS was entered. The prefetched instruction will not be used and INIT will send out a new fetch request. This return from a WCS instruction should be used after jump instructions. No special contents of the sequencer stack are necessary.

PMODX:

The WCS instruction need not have had a WAIT cycle. Except for that, PMODX is equal to INIT. One can easily return to PMODX by using the top word of the stack as return address.

STEND:

This return address should be used from instructions storing words in main memory, if one wishes to use the prefetched word if that is not modified. The WCS instruction must have read the DBR onto IDB with an accompanying WAIT cycle, and saved it in GPR. The ALU register R1 must contain the address written into. If multiple stores are performed by the WCS routine, a microprogram subroutine, WRRQ, can be used to send out new requests and to handle the R = P flip-flop (see STF as an example). If * + 1 is modified, a new fetch is performed, if it is not the prefetched instruction saved in GPR is used. The sequence stack must be popped once before the jump to STEND to bring USDBR to the top word.

Note that it is always safest to end a machine instruction with a jump to PMODX but that it will save memory accesses and time if one of the other return points is used. To find examples, the reader is advised to study the microprogram listing and the flow charts in Appendix E carefully.

Example of a machine instruction "set argument to register":

1	1 0 0	0 0 0	0 1 1	x x x	x x x
1	4	0	3		DR

The argument (0 - 7) is moved into the destination register, which can be any of the registers D, P, B, L, A, T, or X. Since the code for the instruction is 1403xx, which is CUST2, its entry address is 4001. The code for the machine instruction is two microwords, including the test for P as destination. If P is the destination, the next instruction must be fetched again and the instruction returns to PMODX. If P is not the destination, return is to USDBR. The stack is then popped. The displacement is found by using the B operand equal to IR bits 3 - 5 and using BARG as IDB source.

```
4001/
  B, SRCE          ALUF, PASSD      ALUD, Q
IDBS, BARG         T, JMP           T, HOLD        CYCL, FAST
4016;

4016/
  B,DEST           ALUF, PASSQ      ALUD, B
IDBS, ALU          T, RETURN        T, HOLD
USDBR              CONDENABL;
```

5 WRITABLE CONTROL STORE USAGE

The symbolic microprogram will most easily be prepared using the editor QED, with SINTRAN III. It should be saved on an ordinary file.

Thereafter, the NORD-100 WCS assembler is loaded, by typing N-100-WCS-ASSEM↵ to SINTRAN III. The assembly process is started by typing DO 10 to the assembler. The user is asked to give the names of the input file, list file and object output file. The assembly goes on until the input file is exhausted, and an image of the WCS is then saved on the object file. This file will contain 2066 bytes, of which the first 16 and the last 2 are header and trailers, and not WCS microprogram code. The remaining 2048 bytes are the microprogram to be loaded into WCS.

In order to load this microprogram into WCS, the bytes must be placed in physical addresses 36000 to 37777. If the 2048 bytes on the files are numbered from 1 to 2048, the correspondence between main memory addresses and bytes is as follows:

36000	7	8
36001	5	6
36002	3	4
36003	1	2
36004	15	16
36005	13	14
36006	11	12
36007	9	10
37774	2047	2048
37775	2045	2046
37776	2043	2044
37777	2041	2042

The correspondence between main memory and WCS is indicated in Appendix F.

When all the 2048 bytes have been placed according to this scheme, the machine instruction 143500 (LWCS) must be executed. Thereafter, customer specified instructions may be executed. If they are executed before LWCS is performed, a ROM-out-of-range internal interrupt will be generated, and the WCS will not be used. After power up, or after pushing the Master Clear button of the computer, WCS will be unaccessible and LWCS must be executed in order to allow customer specified instructions to be executed.

APPENDIX A

MNEMONIC LIST

```
 1 A,Z          000000 000000 000000 000000 A-OPERAND IS 0. USED TO ADDRESS REGISTER 0, CONTAINING 0, STATUS OR SCRATCH
 2 A,D          000000 000000 000000 010000 A-OPERAND IS 1. USED TO ADDRESS THE D-REGISTER
 3 A,P          000000 000000 000000 020000 A-OPERAND IS 2. USED TO ADDRESS THE P-REGISTER
 4 A,B          000000 000000 000000 030000 A-OPERAND IS 3. USED TO ADDRESS THE B-REGISTER
 5 A,L          000000 000000 000000 040000 A-OPERAND IS 4. USED TO ADDRESS THE L-REGISTER
 6 A,A          000000 000000 000000 050000 A-OPERAND IS 5. USED TO ADDRESS THE A-REGISTER
 7 A,T          000000 000000 000000 060000 A-OPERAND IS 6. USED TO ADDRESS THE T-REGISTER
 8 A,X          000000 000000 000000 070000 A-OPERAND IS 7. USED TO ADDRESS THE X-REGISTER
 9 A,STS        000000 000000 000000 100000 A-OPERAND IS 10. USED TO ADDRESS REGISTER 10, CONTAINING STATUS OR SCRATCH
10 A,R1         000000 000000 000000 110000 A-OPERAND IS 11. USED TO ADDRESS REGISTER 11, CONTAINING ADDRESS OR SCRATCH
11 A,R2         000000 000000 000000 120000 A-OPERAND IS 12. USED TO ADDRESS REGISTER 12, CONTAINING SCRATCH
12 A,R3         000000 000000 000000 130000 A-OPERAND IS 13. USED TO ADDRESS REGISTER 13, CONTAINING SCRATCH
13 A,R4         000000 000000 000000 140000 A-OPERAND IS 14. USED TO ADDRESS REGISTER 14, CONTAINING SCRATCH
14 A,R5         000000 000000 000000 150000 A-OPERAND IS 15. USED TO ADDRESS REGISTER 15, CONTAINING SCRATCH
15 A,R6         000000 000000 000000 160000 A-OPERAND IS 16. USED TO ADDRESS REGISTER 16, CONTAINING SCRATCH
16 A,R7         000000 000000 000000 170000 A-OPERAND IS 17. USED TO ADDRESS REGISTER 17, CONTAINING SCRATCH
17 A,PIL        000010 000000 000000 000000 A-OPERAND IS THE VALUE IN THE 'PIL'-REGISTER, LOADED BY 'COMM,LDPIL'
18 A,REG        000020 000000 000000 000000 A-OPERAND IS THE VALUE OF 'INSTRUCTION BITS 3-6' LOADED BY LAST 'T,MAPJ'
19 A,SRCE       000024 000000 000000 000000 A-OPERAND IS THE VALUE OF 'INTRUCTION BITS 3-5' LOADED BY LAST 'T,MAPJ'
20 A,LC         000030 000000 000000 000000 A-OPERAND IS THE  VALUE OF THE 'LOOP COUNTER' TWO MICROINSTRUCTIONS AGO
21 PIC,RSTS     000000 000000 000000 060000 A-OPERAND IS 6. USED TO GIVE 'READ STATUS'-COMMAND TO INTERRUPT HARDWARE
22 PIC,MCL      000000 000000 000000 000000 A-OPERAND IS 0. USED TO GIVE 'MASTER CLEAR'-COMMAND TO INTERRUPT HARDWARE
23 PIC,MCLPID   000000 000000 000000 020000 A-OPERAND IS 2. USED TO GIVE 'MASKED CLEAR PID'-COMMAND TO INTERRUPT HARDWARE
24 PIC,RMSK     000000 000000 000000 070000 A-OPERAND IS 7. USED TO GIVE 'READ MASK'-COMMAND TO INTERRUPT HARDWARE
25 PIC,LOSTS    000000 000000 000000 110000 A-OPERAND IS 11. USED TO GIVE 'LOAD STATUS'-COMMAND TO INTERRUPT HARDWARE
26 PIC,LMSK     000000 000000 000000 160000 A-OPERAND IS 16. USED TO GIVE 'LOAD MASK'-COMMAND TO INTERRUPT HARDWARE
27 PIC,MCLMSK   000000 000000 000000 120000 A-OPERAND IS 12. USED TO GIVE 'MASKED CLEAR MASK'-COMMAND TO INTERRUPT HARDWARE
28 PIC,IOF      000000 000000 000000 150000 A-OPERAND IS 15. USED TO GIVE 'IOF'-COMMAND TO INTERRUPT HARDWARE
29 PIC,ION      000000 000000 000000 170000 A-OPERAND IS 17. USED TO GIVE 'ION'-COMMAND TO INTERRUPT HARDWARE
30 PIC,MSTMSK   000000 000000 000000 130000 A-OPERAND IS 13. USED TO GIVE 'MASKED SET MASK'-COMMAND TO INTERRUPT HARDWARE
31 PIC,RVECT    000000 000000 000000 050000 A-OPERAND IS 5. USED TO GIVE 'READ-VECTOR'-COMMAND TO INTERRUPT HARDWARE
32 A,0          000000 000000 000000 000000 A-OPERAND IS 0
33 A,1          000000 000000 000000 010000 A-OPERAND IS 1
34 A,2          000000 000000 000000 020000 A-OPERAND IS 2
35 A,3          000000 000000 000000 030000 A-OPERAND IS 3
36 A,4          000000 000000 000000 040000 A-OPERAND IS 4
37 A,5          000000 000000 000000 050000 A-OPERAND IS 5
38 A,6          000000 000000 000000 060000 A-OPERAND IS 6
39 A,7          000000 000000 000000 070000 A-OPERAND IS 7
40 A,10         000000 000000 000000 100000 A-OPERAND IS 10
41 A,11         000000 000000 000000 110000 A-OPERAND IS 11
42 A,12         000000 000000 000000 120000 A-OPERAND IS 12
43 A,13         000000 000000 000000 130000 A-OPERAND IS 13
44 A,14         000000 000000 000000 140000 A-OPERAND IS 14
45 A,15         000000 000000 000000 150000 A-OPERAND IS 15
46 A,16         000000 000000 000000 160000 A-OPERAND IS 16
47 A,17         000000 000000 000000 170000 A-OPERAND IS 17
48 B,Z          000000 000000 000000 000000 B-OPERAND IS 0. USED TO ADDRESS REGISTER 0, CONTAINING 0, STATUS OR SCRATCH
49 B,D          000000 000000 000001 000000 B-OPERAND IS 1. USED TO ADDRESS THE D-REGISTER
50 B,P          000000 000000 000002 000000 B-OPERAND IS 2. USED TO ADDRESS THE P-REGISTER
51 B,B          000000 000000 000003 000000 B-OPERAND IS 3. USED TO ADDRESS THE B-REGISTER
52 B,L          000000 000000 000004 000000 B-OPERAND IS 4. USED TO ADDRESS THE L-REGISTER
53 B,A          000000 000000 000005 000000 B-OPERAND IS 5. USED TO ADDRESS THE A-REGISTER
54 B,T          000000 000000 000006 000000 B-OPERAND IS 6. USED TO ADDRESS THE T-REGISTER
55 B,X          000000 000000 000007 000000 B-OPERAND IS 7. USED TO ADDRESS THE X-REGISTER
56 B,STS        000000 000000 000010 000000 B-OPERAND IS 10. USED TO ADDRESS REGISTER 10, CONTAINING STATUS OR SCRATCH
57 B,R1         000000 000000 000011 000000 B-OPERAND IS 11. USED TO ADDRESS REGISTER 11, CONTAINING ADDRESS OR SCRATCH
58 B,R2         000000 000000 000012 000000 B-OPERAND IS 12. USED TO ADDRESS REGISTER 12, CONTAINING SCRATCH
59 B,R3         000000 000000 000013 000000 B-OPERAND IS 13. USED TO ADDRESS REGISTER 13, CONTAINING SCRATCH
60 B,R4         000000 000000 000014 000000 B-O    ND IS 14. USED TO ADDRESS REGISTER 14, CONTAINING S    TCH
```

```
 61 B,R5          000000 000000 000015 000000 B-OPERAND IS 15. USED TO ADDRESS REGISTER 15, CONTAINING SCRATCH
 62 B,R6          000000 000000 000016 000000 B-OPERAND IS 16. USED TO ADDRESS REGISTER 16, CONTAINING SCRATCH
 63 B,R7          000000 000000 000017 000000 B-OPERAND IS 17. USED TO ADDRESS REGISTER 17, CONTAINING SCRATCH
 64 B,DEST        000001 000000 000000 000000 B-OPERAND IS THE VALUE OF 'INSTRUCTION BITS 0-2' LOADED BY LAST 'T,MAPJ'
 65 B,SRCE        000002 000000 000000 000000 B-OPERAND IS THE VALUE OF 'INSTRUCTION BITS 3-5' LOADED BY LAST 'T,MAPJ'
 66 B,LC          000003 000000 000000 000000 B-OPERAND IS THE VALUE IF THE 'LOOP COUNTER' TWO INSTRUCTIONS AGO
 67 B,0           000000 000000 000000 000000 B-OPERAND IS 0
 68 B,1           000000 000000 000001 000000 B-OPERAND IS 1
 69 B,2           000000 000000 000002 000000 B-OPERAND IS 2
 70 B,3           000000 000000 000003 000000 B-OPERAND IS 3
 71 B,4           000000 000000 000004 000000 B-OPERAND IS 4
 72 B,5           000000 000000 000005 000000 B-OPERAND IS 5
 73 B,6           000000 000000 000006 000000 B-OPERAND IS 6
 74 B,7           000000 000000 000007 000000 B-OPERAND IS 7
 75 B,10          000000 000000 000010 000000 B-OPERAND IS 10
 76 B,11          000000 000000 000011 000000 B-OPERAND IS 11
 77 B,12          000000 000000 000012 000000 B-OPERAND IS 12
 78 B,13          000000 000000 000013 000000 B-OPERAND IS 13
 79 B,14          000000 000000 000014 000000 B-OPERAND IS 14
 80 B,15          000000 000000 000015 000000 B-OPERAND IS 15
 81 B,16          000000 000000 000016 000000 B-OPERAND IS 16
 82 B,17          000000 000000 000017 000000 B-OPERAND IS 17
 83 ALUF,ANDAQ    010000 000000 000000 000000 A /\ Q -> F
 84 ALUF,ANDAB    010200 000000 000000 000000 A /\ B -> F
 85 ALUF,ANDDA    011200 000000 000000 000000 D /\ A -> F
 86 ALUF,ANDDQ    011400 000000 000000 000000 D /\ Q -> F
 87 ALUF,ORAQ     006000 000000 000000 000000 A \/ Q -> F
 88 ALUF,ORAB     006200 000000 000000 000000 A \/ B -> F
 89 ALUF,ORDA     007200 000000 000000 000000 D \/ A -> F
 90 ALUF,ORDQ     007400 000000 000000 000000 D \/ Q -> F
 91 ALUF,XORAQ    014000 000000 000000 000000 A XOR Q -> F
 92 ALUF,XORAB    014200 000000 000000 000000 A XOR B -> F
 93 ALUF,XORDA    015200 000000 000000 000000 D XOR A -> F
 94 ALUF,XORDQ    015400 000000 000000 000000 D XOR Q -> F
 95 ALUF,XNORAQ   016000 000000 000000 000000 NOT(A XOR Q) -> F
 96 ALUF,XNORAB   016200 000000 000000 000000 NOT(A XOR B) -> F
 97 ALUF,XNORDA   017200 000000 000000 000000 NOT(D XOR A) -> F
 98 ALUF,XNORDQ   017400 000000 000000 000000 NOT(D XOR Q) -> F
 99 ALUF,ZERO     010400 000000 000000 000000 0 -> F
100 ALUF,INVQ     016400 000000 000000 000000 NOT(Q) -> F
101 ALUF,INVB     016600 000000 000000 000000 NOT(B) -> F
102 ALUF,INVA     017000 000000 000000 000000 NOT(A) -> F
103 ALUF,INVD     017600 000000 000000 000000 NOT(D) -> F
104 ALUF,PASSQ    014400 000000 000000 000000 Q -> F
105 ALUF,PASSB    014600 000000 000000 000000 B -> F
106 ALUF,PASSA    015000 000000 000000 000000 A -> F
107 ALUF,PASSD    015600 000000 000000 000000 D -> F
108 ALUF,MASKAQ   012000 000000 000000 000000 NOT(A) /\ Q -> F
109 ALUF,MASKAB   012200 000000 000000 000000 NOT(A) /\ B -> F
110 ALUF,MASKDA   013200 000000 000000 000000 NOT(D) /\ A -> F
111 ALUF,MASKDQ   013400 000000 000000 000000 NOT(D) /\ Q -> F
112 ALUF,A+Q      000000 000000 000000 000000 A + Q -> F
113 ALUF,A+B      000200 000000 000000 000000 A + B -> F
114 ALUF,D+A      001200 000000 000000 000000 D + A -> F
115 ALUF,D+Q      001400 000000 000000 000000 D + Q -> F
116 ALUF,A+Q+1    000000 040000 000000 000000 A + Q + 1 -> F
117 ALUF,A+B+1    000200 040000 000000 000000 A + B + 1 -> F
118 ALUF,D+A+1    001200 040000 000000 000000 D + A + 1 -> F
119 ALUF,D+Q+1    001400 040000 000000 000000 D + Q + 1 -> F
120 ALUF,-Q-1     004400 000000 000000 000000 -Q - 1 -> F
```

```
121 ALUF,-B-1       004600 000000 000000 000000 -B - 1 -> F
122 ALUF,-A-1       005000 000000 000000 000000 -A - 1 -> F
123 ALUF,-D-1       003600 000000 000000 000000 -D - 1 -> F
124 ALUF,Q-A-1      002000 000000 000000 000000 Q - A - 1 -> F
125 ALUF,B-A-1      002200 000000 000000 000000 B - A - 1 -> F
126 ALUF,A-D-1      003200 000000 000000 000000 A - D - 1 -> F
127 ALUF,Q-D-1      003400 000000 000000 000000 Q - D - 1 -> F
128 ALUF,A-Q-1      004000 000000 000000 000000 A - Q - 1 -> F
129 ALUF,A-B-1      004200 000000 000000 000000 A - B - 1 -> F
130 ALUF,D-A-1      005200 000000 000000 000000 D - A - 1 -> F
131 ALUF,D-Q-1      005400 000000 000000 000000 D - Q - 1 -> F
132 ALUF,Q          000400 000000 000000 000000 Q -> F
133 ALUF,B          000600 000000 000000 000000 B -> F
134 ALUF,A          001000 000000 000000 000000 A -> F
135 ALUF,D          001600 000000 000000 000000 D -> F
136 ALUF,Q+1        000400 040000 000000 000000 Q + 1 -> F
137 ALUF,B+1        000600 040000 000000 000000 B + 1 -> F
138 ALUF,A+1        001000 040000 000000 000000 A + 1 -> F
139 ALUF,D+1        001600 040000 000000 000000 D + 1 -> F
140 ALUF,Q-1        002400 000000 000000 000000 Q - 1 -> F
141 ALUF,B-1        002600 000000 000000 000000 B - 1 -> F
142 ALUF,A-1        003000 000000 000000 000000 A - 1 -> F
143 ALUF,D-1        005600 000000 000000 000000 D - 1 -> F
144 ALUF,-Q         004400 040000 000000 000000 -Q -> F
145 ALUF,-B         004600 040000 000000 000000 -B -> F
146 ALUF,-A         005000 040000 000000 000000 -A -> F
147 ALUF,-D         003600 040000 000000 000000 -D -> F
148 ALUF,Q-A        002000 040000 000000 000000 Q - A -> F
149 ALUF,B-A        002200 040000 000000 000000 B - A -> F
150 ALUF,A-D        003200 040000 000000 000000 A - D -> F
151 ALUF,Q-D        003400 040000 000000 000000 Q - D -> F
152 ALUF,A-Q        004000 040000 000000 000000 A - Q -> F
153 ALUF,A-B        004200 040000 000000 000000 A - B -> F
154 ALUF,D-A        005200 040000 000000 000000 D - A -> F
155 ALUF,D-Q        005400 040000 000000 000000 D - Q -> F
156 ALUD,Q          000000 000000 000000 000000 F -> Q ; F -> Y
157 ALUD,NONE       020000 000000 000000 000000 F -> Y
158 ALUD,B,YA       040000 000000 000000 000000 F -> B ; A -> Y
159 ALUD,B          060000 000000 000000 000000 F -> B ; F -> Y
160 ALUD,SRD        100000 000000 000000 000000 F -> Y ; (F,Q)/2 -> (B,Q)
161 ALUD,SRB        120000 000000 000000 000000 F -> Y ; F/2 -> B
162 ALUD,SLD        140000 000000 000000 000000 F -> Y ; (F,Q)*2 -> (B,Q)
163 ALUD,SLB        160000 000000 000000 000000 F -> Y ; F*2 -> B
164 STS,LO          000140 000000 000000 000000 IDB-BITS 0-7 -> STATUS-BITS 0-7
165 STS,EA          000040 000000 000000 000000 ALU-CRY -> STATUS C ; ALU-OVF -> STATUS Q ; ALU-OVF \/ STATUS 0 -> STATUS 0
166 STS,ES          000100 000000 000000 000000 ALU SHIFT OUTPUT -> STATUS M
167 CRY,C           000000 100000 000000 000000 STATUS C -> CARRY IN
168 CRY,GPR         000000 140000 000000 000000 GPR BIT 0 -> CARRY IN
169 ALUM,FMU        000000 010000 000000 000000 MULTIPLICATION ALU MODE. GPR-BIT 0 -> ALU-INSTR-BIT 1 ; RIGHT GPR-SHIFT
170 ALUM,FDV        000000 020000 000000 000000 DIVISION ALU MODE. GPR-BIT 0 -> ALU-INSTR-BIT 3 ; LEFT GPR-SHIFT
171 ALUM,IR         000000 030000 000000 000000 SHIFT INSTRUCTION ALU MODE. SHIFT MODE FROM IR-BITS. M IS SET AUTOMATICALLY
172 ALUM,MIC        000000 000000 000000 000000 MICROPROGRAM CONTROLLED SHIFT. SHIFT MODE FROM 'MIS-BITS'
173 MIS,FETCH       000000 000000 000000 000000 INDICATES 'FETCH'-REQUEST TO PAGING AND PROTECTION HARDWARE
174 MIS,PT          000000 002000 000000 000000 INDICATES NORMAL PAGE TABLE TO PAGING HARDWARE
175 MIS,APT         000000 004000 000000 000000 INDICATES ALTERNATIVE PAGE TABLE TO PAGING HARDWARE
176 MIS,HOLD        000000 006000 000000 000000 INDICATES THAT THE LAST USED PAGE TABLE SHOULD BE USED BY THE PAGING HARDWARE
177 MIS,ROT         000000 002000 000000 000000 SPECIFIES ROTATIONAL SHIFT IF 'ALUM,MIC'
178 MIS,ZIN         000000 004000 000000 000000 SPECIFIES ZERO-END-INPUT SHIFT IF 'ALUM,MIC'
179 MIS,LIN         000000 006000 000000 000000 SPECIFIES LINK-END-INPUT SHIFT IF 'ALUM,MIC'
180 CYCL,FAST       000000 000000 000000 000000 SPEC  'ES 'FAST' MICRO INSTRUCTION CYCLE
```

```
181 CYCL,NORM      000000 000000 000020 000000 SPECIFIES 'NORMAL' MICRO INSTRUCTION CYCLE
182 CYCL,SLOW      000000 000000 000040 000000 SPECIFIES 'SLOW' MICRO INSTRUCTION CYCLE
183 CYCL,WAIT      000000 000000 000060 000000 SPECIFIES 'WAIT-FOR-BUS-SYNCHRONIZATION' MICRO CYCLE
184 LCOUNT         000000 000000 000100 000000 COUNT LOOP-COUNTER ; IF FALSE 'RETURN'&'HOLD' ; IF TRUE USE TRUE SPECIFICATIONS
185 CONDENABL      000000 000000 000200 000000 ENABLE CONDITIONAL SEQUENCING, USE 'FALSE' SPECS IF CONDITION FALSE
186 SLWCS          000000 000000 002000 000000 START LOAD WRITABLE CONTROL STORE, AVOID 'ROM-OUT-OF-RANGE'
187 IDBS,ALU       000000 000000 000000 000000 ARITHMETIC-LOGIC-UNIT -> IDB
188 IDBS,BMG       000000 000040 000000 000000 BIT-MASK-GENERATOR -> IDB
189 IDBS,GPR       000000 000100 000000 000000 GENERAL-PURPOSE-REGISTER -> IDB
190 IDBS,DBR       000000 000140 000000 000000 DATA-BUS-REGISTER -> IDB
191 IDBS,ARG       000000 000200 000000 000000 ARGUMENT (MICRO-INSTRUCTION-BITS 0 - 15) -> IDB
192 IDBS,REG       000000 000240 000000 000000 REGISTER-FILE -> IDB
193 IDBS,STS       000000 000300 000000 000000 STATUS -> IDB
194 IDBS,MMR       000000 000340 000000 000000 MEMORY-MANAGEMENT-REGISTER -> IDB
195 IDBS,BARG      000000 000400 000000 000000 B-OPERAND-ARGUMENT (0-17) -> IDB
196 IDBS,SWAP      000000 000440 000000 000000 BYTE-SWAP OF LAST IDB -> IDB
197 IDBS,PEA       000000 000500 000000 000000 PEA-REGISTER -> IDB
198 IDBS,PES       000000 000540 000000 000000 PES-REGISTER -> IDB
199 IDBS,AARG      000000 000600 000000 000000 A-OPERAND-ARGUMENT*10 (0-170) -> IDB
200 IDBS,PIC       000000 000640 000000 000000 PRIORITY-INTERRUPT-CONTROL STATUS REGISTER BUS -> IDB
201 IDBS,IOR       000000 000700 000000 000000 UART-(UNIVERSAL ASYNCHRONOUS RECEIVE TRANSMIT) DATA AND STATUS -> IDB
202 IDBS,DSABL     000000 000740 000000 000000 DISABLE SOURCES TO IDB (USED TO READ 'PIC'-INFO, EXCEPT PIC-STATUS)
203 IDBS,GPR,SEXT  000000 001100 000000 000000 GPR-BITS 0-7 (WITH BITS 8-15 EQUAL TO BIT 7) -> IDB
204 COMM,ESRP      000000 000010 000000 000000 SET 'R=P'-FLIP-FLOP IF 'F=0' IS TRUE
205 COMM,EPIC      000000 000011 000000 000000 A-OPERAND IS AN INSTRUCTION TO 'PIC' (PRIORITY INTERRUPT CONTROLLER)
206 COMM,SMPID     000000 000012 000000 000000 SET MICRO-PID. PID-POSITIONS WHERE IDB HAS A '1' IS FORCED HIGH
207 COMM,START     000000 000013 000000 000000 RESET THE 'STOP'-FLIP-FLOP
208 COMM,SSTOP     000000 000014 000000 000000 SET THE 'STOP'-FLIP-FLOP
209 COMM,CLRTC     000000 000015 000000 000000 CLEAR THE 20 MS CLOCK FLIP-FLOP
210 COMM,CLFF      000000 000016 000000 000000 CLEAR THE 'OOD'- AND THE 'DZD'-FLIP-FLOP
211 COMM,LDLC      000000 000017 000000 000000 LOAD THE 'LOOP COUNTER' WITH THE 6 LOWER IDB-BITS
212 COMM,LDPIL     000000 000001 000000 000000 IDB-BITS 8-15 -> STATUS-BITS 8-15
213 COMM,LDGPR     000000 000002 000000 000000 IDB -> GENERAL-PURPOSE-REGISTER
214 COMM,EWRF      000000 000003 000000 000000 IDB -> REGISTER FILE WORD ADDRESSED BY A-OPERAND AND B-OPERAND
215 COMM,CLIRQ     000000 000004 000000 000000 PREVENT JUMP TO THE INTERRUPT VECTOR, REMOVE 'PANEL'-EFFECT ON 'IRQ'-TEST
216 COMM,RSDA      000000 000005 000000 000000 RESET DATA AVAILABLE IN 'UART'
217 COMM,TBSTR     000000 000006 000000 000000 'TRANSMIT BUFFER STROBE' TO 'UART'. LOWER IDB-BITS LOADED INTO 'UART'
218 COMM,SIOC      000000 000007 000000 000000 LOAD 'I/O-CONTROL' REGISTER, CONTROLLING TERMINAL 1 AND CLOCK INTERRUPTS ETC.
219 COMM,RDRQ      000000 000034 000000 000000 READ MEMORY REQUEST. ADDRESS PRESENT ON IDB
220 COMM,WRRQ      000000 000035 000000 000000 WRITE MEMORY REQUEST. ADDRESS PRESENT ON IDB
221 COMM,IDENT     000000 000036 000000 000000 START 'IDENT'-CYCLE ON BUS. ADDRESS PRESENT ON IDB
222 COMM,IOX       000000 000037 000000 000000 START 'IOX'-CYCLE ON BUS. ADDRESS PRESENT ON BUS
223 COMM,IRDRQ     000000 000030 000000 000000 INDIRECT ADDRESS MEMORY REQUEST. ADDRESS ON IDB
224 COMM,DERQ      000000 000031 000000 000000 DEPOSIT MEMORY REQUEST. ADDRESS ON IDB
225 COMM,EXRQ      000000 000030 000000 000000 EXAMINE MEMORY REQUEST. ADDRESS ON IDB
226 COMM,LDIR      000000 000020 000000 000000 LOAD INT. REG. NO. INTO MMM
227 COMM,XMM       000000 000021 000000 000000 READ/WRITE MMM REGISTER
228 T,JMP          000000 000000 000000 000000 TRUE SEQUENCE IS JMP
229 T,JMP0-3       000000 000000 001000 000000 TRUE SEQUENCE IS JMP. IR-BITS 0-3 CONTROLS THE FOUR LOWER JUMP ADDRESS BITS
230 T,JMP8-10      000000 002000 001000 000000 TRUE SEQUENCE IS JMP. IR-BITS 8-10 CONTROLS THE 3 LOWER JUMP ADDRESS BITS
231 T,JMPALD       000000 004000 001000 000000 TRUE SEQUENCE IS JMP. 'ALD' CONTROLS THE 4 LOWER JUMP ADDRESS BITS
232 T,JMPAOPR      000000 006000 001000 000000 TRUE SEQUENCE IS JMP. THE A-OPERAND CONTROLS THE 4 LOWER JUMP ADDRESS BITS
233 T,MAPJ         000000 000000 004000 000000 IF NEXT INSTRUCTION IS JMP, THE JMP WILL BE MAPPED BY THE MAPPING HARDWARE
234 T,RETURN       000000 000000 040000 000000 TRUE SEQUENCE IS RETURN
235 T,NEXT         000000 000000 100000 000000 TRUE SEQUENCE IS NEXT
236 T,HOLD         000000 000000 000000 000000 TRUE STACK OPERATION IS HOLD
237 T,POP          000000 000000 010000 000000 TRUE STACK OPERATION IS POP
238 T,LOAD         000000 000000 020000 000000 TRUE STACK OPERATION IS LOAD
239 T,PUSH         000000 000000 030000 000000 TRUE STACK OPERATION IS PUSH
240 F,JMP          000000 000000 000000 000000 FALSE SEQUENCE IS JMP. USED TOGETHER WITH A CONDITION SETTING ('COND,...')
```

```
241 F,RETURN      000000 000000 000000 000004 FALSE SEQUENCE IS RETURN. USED TOGETHER WITH A CONDITION SETTING
242 F,NEXT        000000 000000 000000 000010 FALSE SEQUENCE IS NEXT. USED TOGETHER WITH A CONDITION SETTING
243 F,HOLD        000000 000000 000000 000000 FALSE STACK OPERATION IS HOLD. USED TOGETHER WITH A CONDITION SETTING
244 F,POP         000000 000000 000000 000001 FALSE STACK OPERATION IS POP. USED TOGETHER WITH  A CONDITION SETTING
245 F,LOAD        000000 000000 000000 000002 FALSE STACK OPERATION IS LOAD. USED TOGETHER WITH A CONDITION SETTING
246 F,PUSH        000000 000000 000000 000003 FALSE STACK OPERATION IS PUSH. USED TOGETHER WITH A CONDITION SETTING
247 COND,STP      000000 000000 000400 000360 CONDITION FOR TESTING IS  'STOP'
248 COND,OPCTT1   000000 000000 000400 000220 CONDITION FOR TESTING IS IF TT1 IS ON CPU-BOARD
249 COND,OVF      000000 000000 000400 000240 CONDITION FOR TESTING IS 'OVERFLOW' FROM ALU-BIT-SLICES
250 COND,CRY      000000 000000 000400 000260 CONDITION FOR TESTING IS 'CARRY' FROM ALU-BIT-SLICES
251 COND,F11      000000 000000 000400 000300 CONDITION FOR TESTING IS BIT 11 FROM ALU-BIT-SLICES
252 COND,F15      000000 000000 000400 000320 CONDITION FOR TESTING IS BIT 15 FROM ALU-BIT-SLICES
253 COND,F=0      000000 000000 000400 000340 CONDITION FOR TESTING IS 'ALL ZEROS' FROM ALU-BIT-SLICES
254 COND,DR=2     000000 000000 000400 000000 CONDITION FOR TESTING IS 'IR-BITS 0-2' = '010' (DESTINATION FIELD IS 'DP')
255 COND,LC=0     000000 000000 000400 000020 CONDITION FOR TESTING IS 'LOOP-COUNTER'-CONTENT = 0
256 COND,FETCH    000000 000000 000400 000100 CONDITION FOR TESTING IS LAST MEM.REQ. WAS FETCH
257 COND,IRQ      000000 000000 000400 000040 CONDITION FOR TESTING IS 'IRQ'. CHECKS LEV 10-15 IF 'COMM,CLIRQ' HAS BEEN GIVEN
258 COND,RESTR    000000 000000 000400 000060 CONDITION FOR TESTING IS 'RESTRICTED-MODE'. TRUE IF RING 0-1
259 COND,R=P      000000 000000 000400 000200 CONDITION FOR TESTING IS THE R=P-FLIP-FLOP. TRUE IF '(R=P)' OR '(F=0)' IS TRUE
260 COND,OOD      000000 000000 000400 000120 CONDITION FOR TESTING IS THE 'ONE-OUT-DETECT'-FLIP-FLOP
261 COND,DZD      000000 000000 000400 000140 CONDITION FOR TESTING IS THE 'DOUBLE-ZERO-DETECT'-FLIP-FLOP
262 COND,COND     000000 000000 000400 000160 CONDITION FOR TESTING IS THE OUTCOME OF THE LATEST TEST. (USED TO DELAY TESTS)
263 AB,CDIGI      000000 000000 000011 170000 COUNTER CONTROLLING THE NUMBER OF DIGITS IN AN OCTAL NUMBER
264 AB,UPPNR      000000 000000 000012 130000 SCRATCH WORD KEEPING THE UPPER ADDRESS LIMIT IN A MOPC DUMP COMMAND
265 AB,CURNR      000000 000000 000013 130000 SCRATCH WORD KEEPING THE CURRENT ADDRESS IN A MOPC DUMP COMMAND
266 AB,CNT10      000000 000000 000017 170000 COUNTER CONTROLLING THE NUMBER OF OCTAL NUMBERS PER LINE IN A MOPC DUMP COMMAND
267 AB,PRCHR      000000 000000 000016 160000 SCRATCH WORD CONTAINING THE NEXT CHARACTER TO BE WRITTEN BY MOPC
268 AB,TXT1       000000 000000 000014 160000 SCRATCH WORD CONTAINING DISPLAY TEXT
269 AB,TXT2       000000 000000 000013 160000 SCRATCH WORD CONTAINING DISPLAY TEXT
270 AB,SCRAM      000000 000000 000015 140000 SCRATCH WORD CONTAINING SCRAMBLED REPRESENTATION OF LETTERS IN MOPC-INPUT
271 AB,OCTNR      000000 000000 000015 150000 SCRATCH WORD CONTAINING OCTAL NUMBER ASSEMBLED FROM MOPC-INPUT
272 AB,DISPL      000000 000000 000015 160000 TYPE OF RUNNING DISPLAY
273 AB,OCTAD      000000 000000 000013 150000 ADDRESS OF RUNNING DISPLAY
274 AB,OCTA2      000000 000000 000012 160000 WORD TO EXTEND ADDRESS IN OCTAD TO 24 BITS
275 AB,DEPOS      000000 000000 000017 150000 SOME OCTAL DIGIT HAS BEEN WRITTEN SINCE LAST COMMAND WAS TERMINATED
276 AB,DUMPF      000000 000000 000016 150000 SCRATCH WORD INDICATING THAT A DUMP IS IN PROGRESS
277 AB,RONLY      000000 000000 000014 150000 THE EXAMINED REGISTER IS READ-ONLY
278 AB,WRTYP      000000 000000 000012 150000 TYPE OF VARIABLE IN CASE OF DEPOSIT
279 AB,WRADR      000000 000000 000011 150000 ADDRESS OF VARIABLE IS CASE OF DEPOSIT
280 AB,ACTLV      000000 000000 000017 140000 SCRATCH WORD HOLDING 'ACTIVE LEVELS'
281 AB,PVL        000000 000000 000015 170000 SCRATCH WORD HOLDING 'PREVIOUS LEVEL'
282 AB,IIE        000000 000000 000012 170000 SCRATCH WORD HOLDING A 'PIC'-REPRESENTATION OF THE LAST 'IIE'-SETTING
283 AB,PID        000000 000000 000014 170000 SCRATCH WORD HOLDING THE MICROPROGRAM-KNOWN BITS OF THE 'PID'-REGISTER
284 AB,PIE        000000 000000 000013 170000 SCRATCH WORD HOLDING THE 'PIE'-REGISTER
285 AB,STATUS     000000 000000 000017 160000 SCRATCH WORD HOLDING THE LATEST 'COMM,SIOC'-INFORMATION
286 AB,OCTN2      000000 000000 000015 130000 SCRATCH WORD EXPANDING THE 'AB,OCTNR' TO 24 BITS
287 AB,NUMBR      000000 000000 000012 120000 SCRACH WORD HOLDING AN OCTAL NUMBER BEING PRINTED BY MOPC
288 AB,OPR        000000 000000 000011 160000 SCRATCH WORD HOLDING THE 'OPR'-REGISTER VALUE
289 AB,BRKPT      000000 000000 000016 140000 SCRATCH WORD HOLDING BREAKPOINT ADDRESS
290 AB,SINGL      000000 000000 000014 140000 SCRATCH WORD COUNTING SINGLE-INSTRUCTION
291 AB,BPFLG      000000 000000 000013 140000 SCRATCH WORD INDICATING THAT BREAKPOINT IS ON
292 AB,MACL       000000 000000 000011 140000 SCRATCH WORD HOLDING RETRY COUNTER FOR LOAD AFTER MACL
293 AB,LMP        000000 000000 000012 140000 SCRATCH WORD HOLDING THE 'LMP'-REGISTER VALUE
294 AB,EXMOD      000000 000000 000011 130000 SCRATCH WORD HOLDING THE 'EXM'-REGISTER
295 AB,MANIR      000000 000000 000014 130000 SCRATCH WORD HOLDING FLAG FOR MANUAL IR
296 AB,SSAVE      000000 000000 000016 130000 SCRATCH WORD HOLDING STS DURING DECIMAL INTRUCTIONS
297 REMOVED       000000 000000 000000 000000
298 REMOVED       000000 000000 000000 000000
299 REMOVED       000000 000000 000000 000000
300 REMOVED       000000 000000 000000 000000
```

APPENDIX B

DATA FOR AM 2901 BIT SLICE

Taken from "AM2900 Bipolar Microprocessor Family" issued by Advanced Micro Devices, Inc. (to be contacted for more information).

Am2901

Four-Bit Bipolar Microprocessor Slice
Advanced Micro Devices
Bipolar Microprocessor Circuits

DISTINCTIVE CHARACTERISTICS

- Two-address architecture –
  Independent simultaneous access to two working registers saves machine cycles.
- Eight-function ALU –
  Performs addition, two subtraction operations, and five logic functions on two source operands.
- Flexible data source selection –
  ALU data is selected from five source ports for a total of 203 source operand pairs for every ALU function.
- Left/right shift independent of ALU –
  Add and shift operations take only one cycle.
- Four status flags –
  Carry, overflow, zero, and negative.
- Expandable –
  Connect any number of Am2901's together for longer word lengths.
- Microprogrammable –
  Three groups of three bits each for source operand, ALU function, and destination control.

GENERAL DESCRIPTION

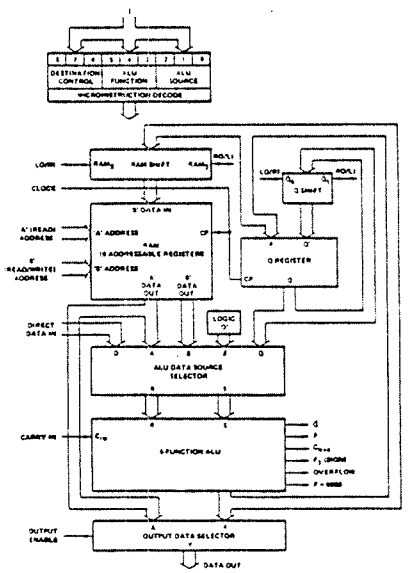
The four-bit bipolar microprocessor slice is designed as a high-speed cascadable element intended for use in CPU's, peripheral controllers, programmable microprocessors and numerous other applications. The microinstruction flexibility of the Am2901 will allow efficient emulation of almost any digital computing machine.

The device, as shown in the block diagram below, consists of a 16-word by 4-bit two-port RAM, a high-speed ALU, and the associated shifting, decoding and multiplexing circuitry. The nine-bit microinstruction word is organized into three groups of three bits each and selects the ALU source operands, the ALU function, and the ALU destination register. The microprocessor is cascadable with full look-ahead or with ripple carry, has three-state outputs, and provides various status flag outputs from the ALU. Advanced low-power Schottky processing is used to fabricate this 40-lead LSI chip.

TABLE OF CONTENTS

Block Diagram	4
Function Tables	5
Package Outline	7
Connection Diagram	8
Pin Definitions	8
Screening	9
Order Codes	9
DC Characteristics	10
AC Characteristics	11
Switching Waveforms	12
Applications	13

MICROPROCESSOR SLICE BLOCK DIAGRAM



ARCHITECTURE

A detailed block diagram of the bipolar microprogrammable microprocessor structure is shown in Figure 1. The circuit is a four-bit slice cascadable to any number of bits. Therefore, all data paths within the circuit are four bits wide. The two key elements in the Figure 1 block diagram are the 16-word by 4-bit 2-port RAM and the high-speed ALU.

Data in any of the 16 words of the Random Access Memory (RAM) can be read from the A-port of the RAM as controlled by the 4-bit A address field input. Likewise, data in any of the 16 words of the RAM as defined by the B address field input can be simultaneously read from the B-port of the RAM. The same code can be applied to the A select field and B select field in which case the identical file data will appear at both the RAM A-port and B-port outputs simultaneously.

When enabled by the RAM write enable (RAM EN), new data is always written into the file (word) defined by the B address field of the RAM. The RAM data input field is driven by a 3-input multiplexer. This configuration is used to shift the ALU output data (F) if desired. This three-input multiplexer scheme allows the data to be shifted up (right) one bit position, shifted down (left) one bit position, or not shifted in either direction.

The RAM A-port data outputs and RAM B-port data outputs drive separate 4-bit latches. These latches hold the RAM data while the clock input is LOW. This eliminates any possible race conditions that could occur while new data is being written into the RAM.

The high-speed Arithmetic Logic Unit (ALU) can perform three binary arithmetic and five logic operations on the two 4-bit input words R and S. The R input field is driven from a 2-input multiplexer, while the S input field is driven from a 3-input multiplexer. Both multiplexers also have an inhibit capability; that is, no data is passed. This is equivalent to a "zero" source operand.

Referring to Figure 1, the ALU R-input multiplexer has the RAM A-port and the direct data inputs (D) connected as inputs. Likewise, the ALU S-input multiplexer has the RAM A-port, the RAM B-port and the Q register connected as inputs.

This multiplexer scheme gives the capability of selecting various pairs of the A, B, D, Q and "0" inputs as source operands to the ALU. These five inputs, when taken two at a time, result in ten possible combinations of source operand pairs. These combinations include AB, AD, AQ, A0, BD, BQ, B0, DQ, D0 and Q0. It is apparent that AD, AQ and A0 are somewhat redundant with BD, BQ and B0 in that if the A address and B address are the same, the identical function results. Thus, there are only seven completely non-redundant source operand pairs for the ALU. The Am2901 microprocessor implements eight of these pairs. The microinstruction inputs used to select the ALU source operands are the I_0, I_1, and I_2 inputs. The definition of I_0, I_1, and I_2 for the eight source operand combinations are as shown in Figure 2. Also shown is the octal code for each selection.

The two source operands not fully described as yet are the D input and Q input. The D input is the four-bit wide direct data field input. This port is used to insert all data into the working registers inside the device. Likewise, this input can be used in the ALU to modify any of the internal data files. The Q register is a separate 4-bit file intended primarily for multiplication and division routines but it can also be used as an accumulator or holding register for some applications.

The ALU itself is a high-speed arithmetic/logic operator capable of performing three binary arithmetic and five logic functions. The I_3, I_4, and I_5 microinstruction inputs are used to select the ALU function. The definition of these inputs is shown in Figure 3. The octal code is also shown for reference. The normal technique for cascading the ALU of several devices is in a look-ahead carry mode. Carry generate, $\overline{G}$, and carry propagate, $\overline{P}$, are outputs of the device for use with a carry-look-ahead-generator such as the Am2902 ('182). A carry-out, C_{n+4}, is also generated and is available as an output for use as the carry flag in a status register. Both carry-in (C_n) and carry-out (C_{n+4}) are active HIGH.

The ALU has three other status-oriented outputs. These are F_3, F = 0, and overflow (OVR). The F_3 output is the most significant (sign) bit of the ALU and can be used to determine positive or negative results without enabling the three-state data outputs. F_3 is non-inverted with respect to the sign bit output Y_3. The F = 0 output is used for zero detect. It is an open-collector output and can be wire OR'ed between microprocessor slices. F = 0 is HIGH when all F outputs are LOW. The overflow output (OVR) is used to flag arithmetic operations that exceed the available two's complement number range. The overflow output (OVR) is HIGH when overflow exists. That is, when C_{n+3} and C_{n+4} are not the same polarity.

The ALU data output is routed to several destinations. It can be a data output of the device and it can also be stored in the RAM or the Q register. Eight possible combinations of ALU destination functions are available as defined by the I_6, I_7, and I_8 microinstruction inputs. These combinations are shown in Figure 4.

The four-bit data output field (Y) features three-state outputs and can be directly bus organized. An output control ($\overline{OE}$) is used to enable the three-state outputs. When $\overline{OE}$ is HIGH, the Y outputs are in the high-impedance state.

A two-input multiplexer is also used at the data output such that either the A-port of the RAM or the ALU outputs (F) are selected at the device Y outputs. This selection is controlled by the I_6, I_7, and I_8 microinstruction inputs. Refer to Figure 4 for the selected output for each microinstruction code combination.

As was discussed previously, the RAM inputs are driven from a three-input multiplexer. This allows the ALU outputs to be entered non-shifted, shifted up one position (X2) or shifted down one position (÷2). The shifter has two ports; one is labeled RAM_0–LO/RI and the other is labeled RAM_3–RO/LI. Both of these ports consist of a buffer-driver with a three-state output and an input to the multiplexer. Thus, in the shift up mode, the RO buffer is enabled and the RI multiplexer input is enabled. Likewise, in the shift down mode, the LO buffer and LI input are enabled. In the no-shift mode, both the LO and RO buffers are in the high-impedance state and the multiplexer inputs are not selected. This shifter is controlled from the I_6, I_7, and I_8 microinstruction inputs as defined in Figure 4.

Similarly, the Q register is driven from a 3-input multiplexer. In the no-shift mode, the multiplexer enters the ALU data into the Q register. In either the shift-up or shift-down mode, the multiplexer selects the Q register data appropriately shifted up or down. The Q shifter also has two ports; one is labeled Q_0–LO/RI and the other is Q_3–RO/LI. The operation of these two ports is similar to the RAM shifter and is also controlled from I_6, I_7, and I_8 as shown in Figure 4.

The clock input to the Am2901 controls the RAM, the Q register, and the A and B data latches. When enabled, data is clocked into the Q register on the LOW-to-HIGH transition of the clock. When the clock input is HIGH, the A and B latches are open and will pass whatever data is present at the RAM outputs. When the clock input is LOW, the latches are closed and will retain the last data entered. If the RAM-EN is enabled, new data will be written into the RAM file (word) defined by the B address field when the clock input is LOW.

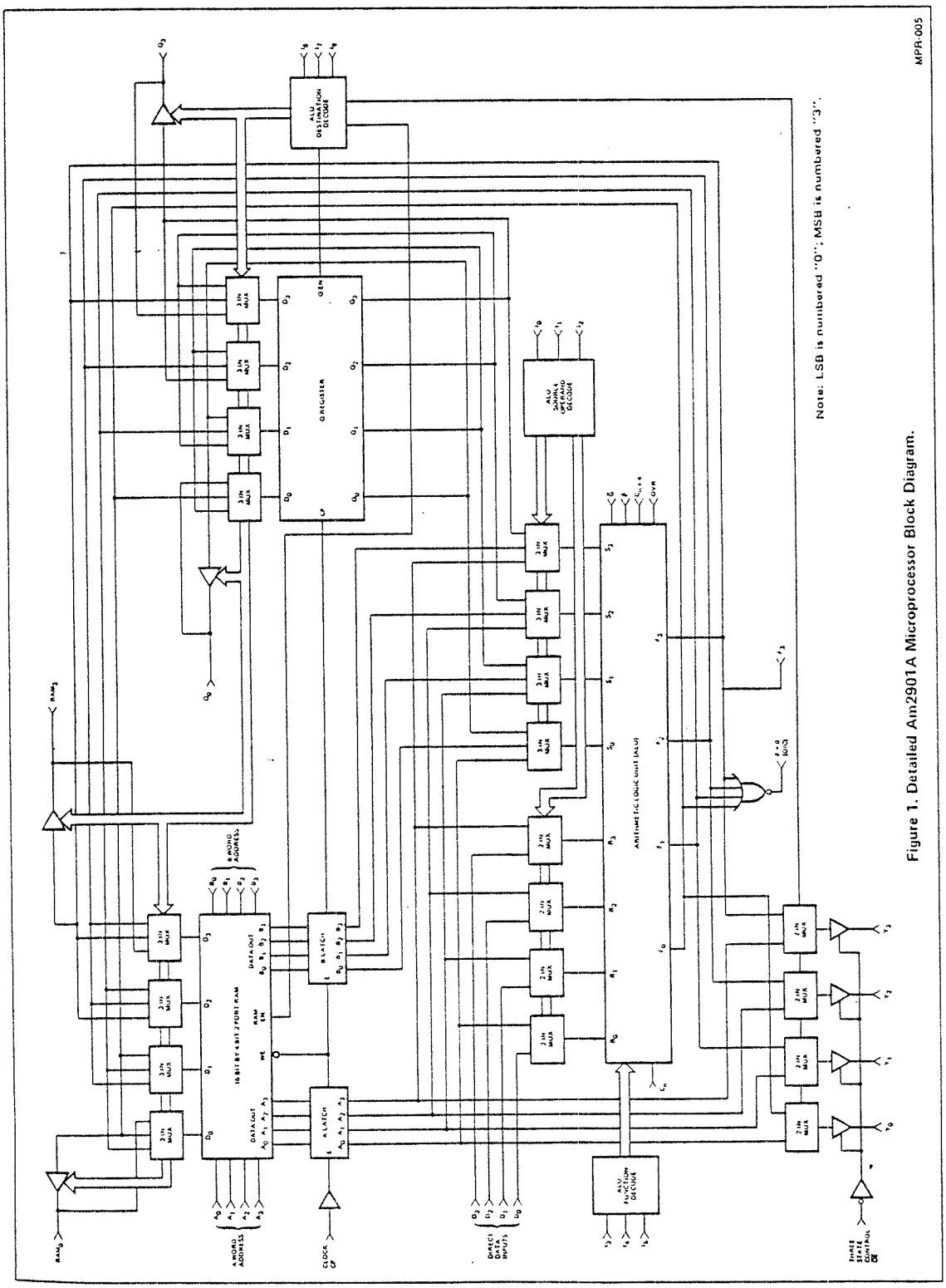


Figure 1. Detailed Am2901A Microprocessor Block Diagram.

MICRO CODE				ALU SOURCE OPERANDS	
I_2	I_1	I_0	Octal Code	R	S
L	L	L	0	A	Q
L	L	H	1	A	B
L	H	L	2	0	Q
L	H	H	3	0	B
H	L	L	4	0	A
H	L	H	5	D	A
H	H	L	6	D	Q
H	H	H	7	0	0

Figure 2. ALU Source Operand Control.

MICRO CODE				ALU Function	Symbol
I_5	I_4	I_3	Octal Code		
L	L	L	0	R Plus S	R + S
L	L	H	1	S Minus R	S – R
L	H	L	2	R Minus S	R – S
L	H	H	3	R OR S	R ∨ S
H	L	L	4	R AND S	R ∧ S
H	L	H	5	$\overline{R}$ AND S	$\overline{R}$ ∧ S
H	H	L	6	R EX-OR S	R ∨ S
H	H	H	7	R EX-NOR S	$\overline{R \vee S}$

Figure 3. ALU Function Control.

MICRO CODE				RAM FUNCTION		Q-REG. FUNCTION		Y OUTPUT	RAM SHIFTER		Q SHIFTER	
I_8	I_7	I_6	Octal Code	Shift	Load	Shift	Load		RAM_0 LO/RI	RAM_3 LI/RO	Q_0 LO/RI	Q_3 LI/RO
L	L	L	0	—	—	NONE	ALU (F_i)	F	X	X	X	X
L	L	H	1	—	—	—	—	F	X	X	X	X
L	H	L	2	NONE	ALU (F_i)	—	—	A	X	X	X	X
L	H	H	3	NONE	ALU (F_i)	—	—	F	X	X	X	X
H	L	L	4	LEFT (DOWN)	ALU (F_{i+1})	LEFT (DOWN)	Q-REG (Q_{i+1})	F	F_0	IN_3	Q_0	IN_3
H	L	H	5	LEFT (DOWN)	ALU (F_{i+1})	—	—	F	F_0	IN_3	Q_0	X
H	H	L	6	RIGHT (UP)	ALU (F_{i-1})	RIGHT (UP)	Q-REG (Q_{i-1})	F	IN_0	F_3	IN_0	Q_3
H	H	H	7	RIGHT (UP)	ALU (F_{i-1})	—	—	F	IN_0	F_3	X	Q_3

X = Don't care. Electrically, the shift pin is a TTL input internally connected to a three-state output which is in the high-impedance state.

Figure 4. ALU Destination Control.

OCTAL I_{543}	OCTAL I_{210}	0	1	2	3	4	5	6	7
	ALU Source / ALU Function	A, Q	A, B	0, Q	0, B	0, A	0, A	0, Q	0, 0
0	C_n = L R Plus S C_n = H	A+Q A+Q+1	A+B A+B+1	Q Q+1	B B+1	A A+1	0+A 0+A+1	0+Q 0+Q+1	0 0+1
1	C_n = L S Minus R C_n = H	Q–A–1 Q–A	B–A–1 B–A	Q–1 Q	B–1 B	A–1 A	A–0–1 A–0	Q–0–1 Q–0	–0–1 –0
2	C_n = L R Minus S C_n = H	A–Q–1 A–Q	A–B–1 A–B	–Q–1 –Q	–B–1 –B	–A–1 –A	0–A–1 0–A	0–Q–1 0–Q	0–1 0
3	R OR S	A∨Q	A∨B	Q	B	A	0∨A	0∨Q	0
4	R AND S	A∧Q	A∧B	0	0	0	0∧A	0∧Q	0
5	$\overline{R}$ AND S	$\overline{A}$∧Q	$\overline{A}$∧B	Q	B	A	$\overline{0}$∧A	$\overline{0}$∧Q	0
6	R EX-OR S	A∨Q	A∨B	Q	B	A	0∨A	0∨Q	0
7	R EX-NOR S	$\overline{A \vee Q}$	$\overline{A \vee B}$	$\overline{Q}$	$\overline{B}$	$\overline{A}$	$\overline{0 \vee A}$	$\overline{0 \vee Q}$	$\overline{0}$

+ = Plus; – = Minus; ∨ = OR; ∧ = AND; ∨ = EX-OR

Figure 5. Source Operand and ALU Function Matrix.

SOURCE OPERANDS AND ALU FUNCTIONS

There are eight source operand pairs available to the ALU as selected by the I_0, I_1, and I_2 instruction inputs. The ALU can perform eight functions; five logic and three arithmetic. The I_3, I_4, and I_5 instruction inputs control this function selection. The carry input, C_n, also affects the ALU results when in the arithmetic mode. The C_n input has no effect in the logic mode. When I_0 through I_5 and C_n are viewed together, the matrix of Figure 5 results. This matrix fully defines the ALU/source operand function for each state.

The ALU functions can also be examined on a "task" basis, i.e., add, subtract, AND, OR, etc. In the arithmetic mode, the carry will affect the function performed while in the logic mode, the carry will have no bearing on the ALU output. Figure 6 defines the various logic operations that the Am2901 can perform and Figure 7 shows the arithmetic functions of the device. Both carry-in LOW (C_n = 0) and carry-in HIGH (C_n = 1) are defined in these operations.

Octal I_{543} I_{210}	Group	Function
4 0	AND	A∧Q
4 1		A∧B
4 5		0∧A
4 6		0∧Q
3 0	OR	A∨Q
3 1		A∨B
3 5		0∨A
3 6		0∨Q
6 0	EX-OR	A∨Q
6 1		A∨B
6 5		0∨A
6 6		0∨Q
7 0	EX-NOR	$\overline{A \vee Q}$
7 1		$\overline{A \vee B}$
7 5		$\overline{0 \vee A}$
7 6		$\overline{0 \vee Q}$
7 2	INVERT	$\overline{Q}$
7 3		$\overline{B}$
7 4		$\overline{A}$
7 7		$\overline{0}$
6 2	PASS	Q
6 3		B
6 4		A
6 7		0
3 2	PASS	Q
3 3		B
3 4		A
3 7		0
4 2	"ZERO"	0
4 3		0
4 4		0
4 7		0
5 0	MASK	$\overline{A} \wedge Q$
5 1		$\overline{A} \wedge B$
5 5		$\overline{0} \wedge A$
5 6		$\overline{0} \wedge Q$

Figure 6. ALU Logic Mode Functions. (C_n Irrelevant)

Octal	C_n = 0 (Low)		C_n = 1 (High)	
I_{543} I_{210}	Group	Function	Group	Function
0 0	ADD	A+Q	ADD plus one	A+Q+1
0 1		A+B		A+B+1
0 5		0+A		0+A+1
0 6		0+Q		0+Q+1
0 2	PASS	Q	Increment	Q+1
0 3		B		B+1
0 4		A		A+1
0 7		0		0+1
1 2	Decrement	Q–1	PASS	Q
1 3		B–1		B
1 4		A–1		A
2 7		0–1		0
2 2	1's Comp.	–Q–1	2's Comp. (Negate)	–Q
2 3		–B–1		–B
2 4		–A–1		–A
1 7		–0–1		–0
1 0	Subtract (1's Comp)	Q–A–1	Subtract (2's Comp)	Q–A
1 1		B–A–1		B–A
1 5		A–0–1		A–0
1 6		Q–0–1		Q–0
2 0		A–Q–1		A–Q
2 1		A–B–1		A–B
2 5		0–A–1		0–A
2 6		0–Q–1		0–Q

Figure 7. ALU Arithmetic Mode Functions.

LOGIC FUNCTIONS FOR G, P, C_{n+4}, AND OVR

The four signals G, P, C_{n+4}, and OVR are designed to indicate carry and overflow conditions when the Am2901 is in the add or subtract mode. The table below indicates the logic equations for these four signals for each of the eight ALU functions. The R and S inputs are the two inputs selected according to Figure 2.

Definitions (+ = OR)

$P_0 = R_0 + S_0$ $\quad G_0 = R_0S_0$

$P_1 = R_1 + S_1$ $\quad G_1 = R_1S_1$

$P_2 = R_2 + S_2$ $\quad G_2 = R_2S_2$

$P_3 = R_3 + S_3$ $\quad G_3 = R_3S_3$

$$C_4 = G_3 + P_3G_2 + P_3P_2G_1 + P_3P_2P_1G_0 + P_3P_2P_1P_0C_n$$

$$C_3 = G_2 + P_2G_1 + P_2P_1G_0 + P_2P_1P_0C_n$$

I_{543}	Function	$\overline{P}$	$\overline{G}$	C_{n+4}	OVR
0	R + S	$\overline{P_3P_2P_1P_0}$	$\overline{G_3 + P_3G_2 + P_3P_2G_1 + P_3P_2P_1G_0}$	C_4	$C_3 \forall C_4$
1	S – R	← Same as R + S equations, but substitute $\overline{R_i}$ for R_i in definitions →			
2	R – S	← Same as R + S equations, but substitute $\overline{S_i}$ for S_i in definitions →			
3	R ∨ S	LOW	$P_3P_2P_1P_0$	$\overline{P_3P_2P_1P_0} + C_n$	$\overline{P_3P_2P_1P_0} + C_n$
4	R ∧ S	LOW	$\overline{G_3 + G_2 + G_1 + G_0}$	$G_3 + G_2 + G_1 + G_0 + C_n$	$\overline{G_3 + G_2 + G_1 + G_0} + \overline{C_n}$
5	$\overline{R} \wedge S$	LOW	← Same as R ∧ S equations, but substitute $\overline{R_i}$ for R_i in definitions →		
6	R ∀ S	← Same as $\overline{R \forall S}$, but substitute $\overline{R_i}$ for R_i in definitions →			
7	$\overline{R \forall S}$	$G_3 + G_2 + G_1 + G_0$	$P_3G_3 + P_3P_2G_2 + P_3P_2P_1G_1 + P_3P_2P_1P_0$	$P_3G_3 + P_3P_2G_2 + P_3P_2P_1G_1 + P_3P_2P_1P_0 (G_0 + \overline{C_n})$	Complement of C_{n+4} at left

+ = OR

Figure 8.

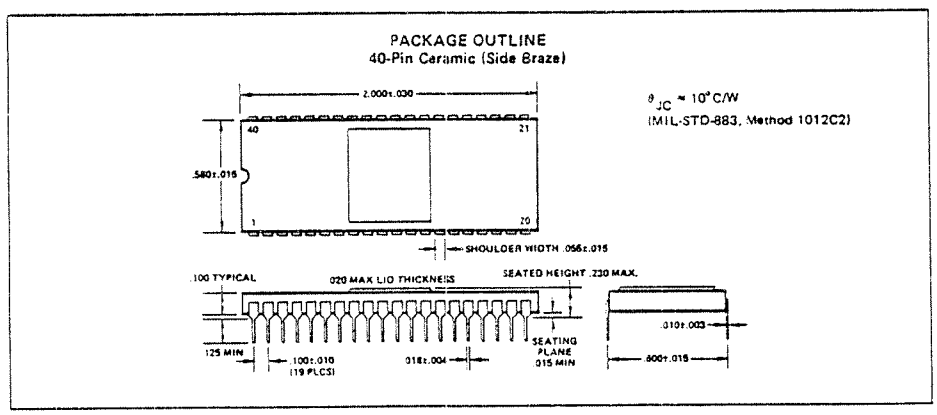


Figure 9.

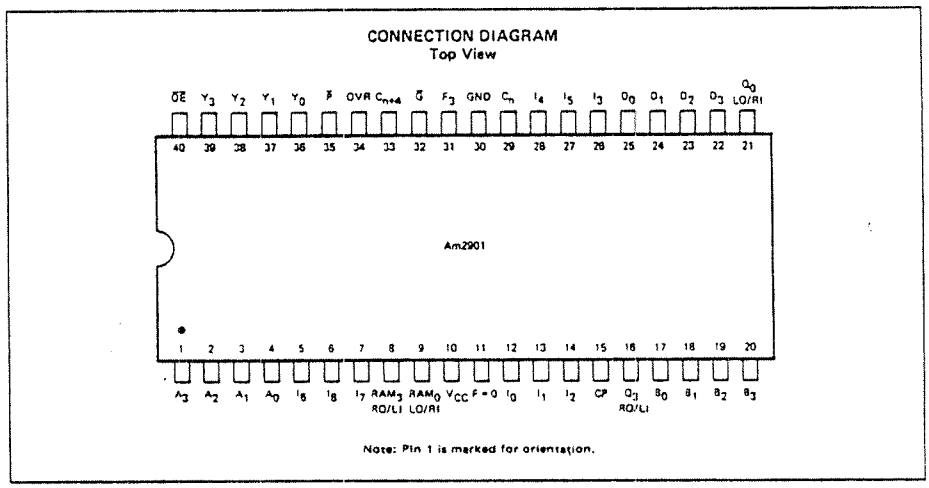


Figure 10.

PIN DEFINITIONS

A_{0-3} The four address inputs to the register stack used to select one register whose contents are displayed through the A-port.

B_{0-3} The four address inputs to the register stack used to select one register whose contents are displayed through the B-port and into which new data can be written when the clock goes LOW.

I_{0-8} The nine instruction control lines to the Am2901, used to determine what data sources will be applied to the ALU (I_{012}), what function the ALU will perform (I_{345}), and what data is to be deposited in the Q-register or the register stack (I_{678}).

RO/LI A shift line at the MSB of the Q register (Q_3RO/LI) and the register stack (RAM_3RO/LI). Electrically these lines are three-state outputs connected to TTL inputs internal to the Am2901. When the destination code on I_{678} indicates a right shift (octal 6 or 7) the three-state outputs are enabled and the MSB of the Q register is available on the Q_3RO/LI pin and the MSB of the ALU output is available on the RAM_3RO/LI pin. Otherwise, the three-state outputs are OFF (high-impedance) and the pins are electrically LS-TTL inputs. When the destination code calls for a down (left) shift, the pins are used as the data inputs to the MSB of the Q register (octal 4) and RAM (octal 4 or 5).

LO/RI Shift lines like RO/LI, but at the LSB of the Q-register and RAM. These pins are tied to the RO/LI pin of the adjacent device to transfer data between devices for left and right shifts of the Q register and ALU data.

D_{0-3} Direct data inputs. A four-bit data field which may be selected as one of the ALU data sources for entering data into the Am2901. D_0 is the LSB.

Y_{0-3} The four data outputs of the Am2901. These are three-state output lines. When enabled, they display either the four outputs of the ALU or the data on the A-port of the register stack, as determined by the destination code I_{678}.

$\overline{OE}$ Output Enable. When $\overline{OE}$ is HIGH, the Y outputs are OFF; when $\overline{OE}$ is LOW, the Y outputs are active (HIGH or LOW).

$\overline{P}, \overline{G}$ The carry generate and propagate outputs of the Am2901's ALU. These signals are used with the Am2902 for carry-lookahead. See Figure 8 for the logic equations.

OVR Overflow. This pin is logically the Exclusive-OR of the carry-in and carry-out of the MSB of the ALU. At the most significant end of the word, this pin indicates that the result of an arithmetic two's complement operation has overflowed into the sign-bit. See Figure 8 for logic equation.

F = 0 This is an open collector output which goes HIGH (OFF) if the data on the four ALU outputs F_{0-3} are all LOW. In positive logic, it indicates the result of an ALU operation is zero.

C_n The carry-in to the Am2901's ALU.

C_{n+4} The carry-out of the Am2901's ALU. See Figure 8 for equations.

CP The clock to the Am2901. The Q register and register stack outputs change on the clock LOW-to-HIGH transition. The clock LOW time is internally the write enable to the 16 x 4 RAM which comprises the "master" latches of the register stack. While the clock is LOW, the "slave" latches on the RAM outputs are closed, storing the data previously on the RAM outputs. This allows synchronous master-slave operation of the register stack.

MAXIMUM RATINGS (Above which the useful life may be impaired)

Storage Temperature	–65°C to +150°C
Temperature (Ambient) Under Bias	–55°C to +125°C
Supply Voltage to Ground Potential	–0.5 V to +6.3 V
DC Voltage Applied to Outputs for HIGH Output State	–0.5 V to +V_{CC} max.
DC Input Voltage	–0.5 V to +5.5 V
DC Output Current, Into Outputs	30 mA
DC Input Current	–30 mA to +5.0 mA

OPERATING RANGE

P/N	Ambient Temperature	V_{CC}
Am2901PC, DC	0°C to +70°C	4.75 V to 5.25 V
Am2901DM, FM	–65°C to +125°C	4.50 V to 5.50 V

STANDARD SCREENING

(Conforms to MIL-STD-883 for Class C Parts)

Step	MIL-STD-883 Method	Conditions	Level Am2901PC, DC	Level Am2901DM, FM
Pre-Seal Visual Inspection	2010	B	100%	100%
Stabilization Bake	1008	C 24-hour 150°C	100%	100%
Temperature Cycle	1010	C –65°C to +150°C 10 cycles	100%	100%
Centrifuge	2001	B 10,000 G	100% *	100%
Fine Leak	1014	A 5×10^{-8} atm-cc/cm³	100% *	100%
Gross Leak	1014	C2 Fluorocarbon	100% *	100%
Electrical Test Subgroups 1 and 7	5004	See below for definitions of subgroups	100%	100%
Insert Additional Screening here for Class B Parts				
Group A Sample Tests	5005	See below for definitions of subgroups		
Subgroup 1			LTPD = 5	LTPD = 5
Subgroup 2			LTPD = 7	LTPD = 7
Subgroup 3			LTPD = 7	LTPD = 7
Subgroup 7			LTPD = 7	LTPD = 7
Subgroup 8			LTPD = 7	LTPD = 7
Subgroup 9			LTPD = 7	LTPD = 7

*Not applicable for Am2901PC

ADDITIONAL SCREENING FOR CLASS B PARTS

Step	MIL-STD-883 Method	Conditions	Level Am2901DMB, FMB
Burn-In	1015	D 125°C 160 hours min.	100%
Electrical Test	5004		
Subgroup 1			100%
Subgroup 2			100%
Subgroup 3			100%
Subgroup 7			100%
Subgroup 9			100%
Return to Group A Tests in Standard Screening			

ORDERING INFORMATION

Package Type	Temperature Range	Order Number
Molded DIP	0°C to +70°C	AM2901PC
Hermetic DIP	0°C to +70°C	AM2901DC
Hermetic DIP	–55°C to +125°C	AM2901DM
Hermetic Flat Pack	–55°C to +125°C	AM2901FM
Dice	0°C to +70°C	AM2901XC

GROUP A SUBGROUPS

(as defined in MIL-STD-883, method 5005)

Subgroup	Parameter	Temperature
1	DC	25°C
2	DC	Maximum rated temperature
3	DC	Minimum rated temperature
7	Function	25°C
8	Function	Maximum and minimum rated temperature
9	Switching	25°C
10	Switching	Maximum Rated Temeperature
11	Switching	Minimum Rated Temperature

ELECTRICAL CHARACTERISTICS OVER OPERATING RANGE (Unless Otherwise Noted)

(Group A, Subgroups 1, 2 and 3)

Parameters	Description	Test Conditions (Note 1)		Min.	Typ. (Note 2)	Max.	Units
V_{OH}	Output HIGH Voltage	V_{CC} = MIN., V_{IN} = V_{IH} or V_{IL}	I_{OH} = −1.6mA Y_0, Y_1, Y_2, Y_3	2.4			Volts
			I_{OH} = −1.0mA, C_{n+4}	2.4			
			I_{OH} = −800µA, OVR, $\overline{P}$	2.4			
			I_{OH} = −600µA, F_3	2.4			
			I_{OH} = −600µA All RO/LI, LO/RI	2.4			
			I_{OH} = −1.6mA, $\overline{G}$	2.4			
I_{CEX}	Output Leakage Current for F = 0 Output	V_{CC} = MIN., V_{OH} = 5.5V V_{IN} = V_{IH} or V_{IL}				250	µA
V_{OL}	Output LOW Voltage	V_{CC} = MIN., V_{IN} = V_{IH} or V_{IL}	I_{OL} = 16mA Y_0, Y_1, Y_2, Y_3			0.5	Volts
			I_{OL} = 10mA, C_{n+4}, F = 0			0.5	
			I_{OL} = 8.0mA, OVR, $\overline{P}$			0.5	
			I_{OL} = 6.0mA, F_3 All RO/LI, LO/RI			0.5	
V_{IH}	Input HIGH Level	Guaranteed input logical HIGH voltage for all inputs		2.0			Volts
V_{IL}	Input LOW Level	Guaranteed input logical LOW voltage for all inputs	Military			0.7	Volts
			Commercial			0.8	
V_I	Input Clamp Voltage	V_{CC} = MIN., I_{IN} = −18mA				−1.5	Volts
I_{IL} (Note 3)	Input LOW Current	V_{CC} = MAX. V_{IN} = 0.5V	Clock, $\overline{OE}$			−0.36	mA
			A_0, A_1, A_2, A_3			−0.36	
			B_0, B_1, B_2, B_3			−0.36	
			D_0, D_1, D_2, D_3			−0.72	
			I_0, I_1, I_2, I_6, I_8			−0.36	
			I_3, I_4, I_5, I_7			−0.72	
			All LO/RI, RO/LI (Note 4)			−0.8	
			C_n			−3.6	
I_{IH} (Note 3)	Input HIGH Current	V_{CC} = MAX. V_{IN} = 2.7V	Clock, $\overline{OE}$			20	µA
			A_0, A_1, A_2, A_3			20	
			B_0, B_1, B_2, B_3			20	
			D_0, D_1, D_2, D_3			40	
			I_0, I_1, I_2, I_6, I_8			20	
			I_3, I_4, I_5, I_7			40	
			All LO/RI, RO/LI (Note 4)			100	
			C_n			200	
I_I	Input HIGH Current	V_{CC} = MAX., V_{IN} = 5.5V				1.0	mA
I_{OZ}	Off State (High Impedance) Output Current	V_{CC} = MAX.	Y_0, Y_1, Y_2, Y_3 V_O = 2.4V			50	µA
			Y_0, Y_1, Y_2, Y_3 V_O = 0.5V			−50	
			All LO/RI, RO/LI V_O = 2.4V (Note 5)			100	
			All LO/RI, RO/LI V_O = 0.5V (Note 5)			−800	
I_{SC}	Output Short Circuit Current (Note 4)		$Y_0, Y_1, Y_2, Y_3, \overline{G}$	−6.0		−40	mA
			C_{n+4}	−6.0		−40	
			OVR, $\overline{P}$	−6.0		−40	
			F_3	−6.0		−40	
			All RO/LI, LO/RI	−6.0		−40	
I_{CC}	Power Supply Current	V_{CC} = MAX.	Military		185	280	mA
			Commercial		185	280	

Notes: 1. For conditions shown as MIN or MAX, use the appropriate value specified under Electrical Characteristics for the applicable device type.
2. Typical limits are at V_{CC} = 5.0V, 25°C ambient and maximum loading.
3. Not more than one output should be shorted at a time. Duration of the short circuit test should not exceed one second.
4. LO/RI and RO/LI are three-state outputs internally connected to TTL inputs. Input characteristics are measured with I_{g7g} in a state such that the three-state output is OFF.

GUARANTEED OPERATING CONDITIONS

Tables I, II, and III below define the timing requirements of the Am2901 in a system. The Am2901 is guaranteed to function correctly over the operating range when used within the delay and set-up time constraints of these tables for the appropriate device type. The tables are divided into three types of parameters; clock characteristics, combinational delays from inputs to outputs, and set-up and hold time requirements. The latter table defines the time prior to the end of the cycle (i.e., clock LOW-to-HIGH transition) that each input must be stable to guarantee that the correct data is written into one of the internal registers.

The performance of the Am2901 within the limits of these tables is guaranteed by the testing defined as "Group A, Subgroup 9" Electrical Testing. For a copy of the tests and limits used for subgroup 9, contact Advanced Micro Devices' Product Marketing.

TABLE I

CYCLE TIME AND CLOCK CHARACTERISTICS

TIME	Am2901DC	Am2901DM
Minimum Read-Modify-Write Cycle (time from selection of A, B registers to end of cycle)	105 ns	120 ns
Maximum Clock Frequency to Shift Q Register (50% duty cycle)	9.5 MHz	8.3 MHz
Minimum Clock LOW Time	30 ns	30 ns
Minimum Clock HIGH Time	30 ns	30 ns
Minimum Clock Period	105 ns	120 ns

TABLE II

MAXIMUM COMBINATIONAL PROPAGATION DELAYS (all in ns, $C_L \leq 15pF$)

From Input \ To Output	Am2901DC								Am2901DM							
	Y	F_3	C_{n+4}	$\overline{G}, \overline{P}$	F=0 R_L=470	OVR	RO, LO RAM	RO, LO Q	Y	F_3	C_{n+4}	$\overline{G}, \overline{P}$	F=0 R_L=470	OVR	RO, LO RAM	RO, LO Q
Clock ↓	115	85	100	100	110	95	105	60	125	95	110	110	120	105	115	65
A, B	110	85	80	80	110	75	110	–	120	95	90	90	120	85	120	–
D	100	70	70	70	100	60	60	–	110	80	75	75	110	65	65	–
C_n	55	35	30	–	50	40	55	–	60	40	30	–	55	45	60	–
I_{012}	85	65	65	65	80	65	80	–	90	70	70	70	85	70	85	–
I_{345}	70	55	60	60	70	60	65	–	75	60	65	65	75	65	70	–
I_{678}	55	–	–	–	–	–	45	45	60	–	–	–	–	–	50	50
$\overline{OE}$ Enable/Disable	40/25	–	–	–	–	–	–	–	40/25	–	–	–	–	–	–	–
A bypassing ALU (I = 2xx)	60	–	–	–	–	–	–	–	65	–	–	–	–	–	–	–

SET-UP AND HOLD TIMES (minimum cycles from each input)

Set-up and hold times are defined relative to the clock LOW-to-HIGH edge. Inputs must be steady at all times from the set-up time prior to the clock until the hold time after the clock. The set-up times allow sufficient time to perform the correct operation on the correct data so that the correct ALU data can be written into one of the registers.

TABLE III

Set-Up and Hold Times (all in ns) (Note 1)

From Input	Notes	Am2901DC		Am2901DM	
		Set-Up Time	Hold Time	Set-Up Time	Hold Time
A, B Source	2, 3, 4	105 t_{pwL} + 30	0	120 t_{pwL} + 30	0
B Dest.	2, 4	t_{pwL} + 15	0	t_{pwL} + 15	0
D		100	0	110	0
C_n		55	0	60	0
I_{012}		85	0	90	0
I_{345}		70	0	75	0
I_{678}	4	t_{pwL} + 15	0	t_{pwL} + 15	0
RI, LI (RAM or Q)		30	0	30	0

Notes: 1. See Figure 11 and 12.
2. If the B address is used as a source operand, allow for the "A, B source" set-up time; if it is used only for the destination address, use the "B dest." set-up time.
3. Where two numbers are shown, both must be met.
4. "t_{pwL}" is the clock LOW time.

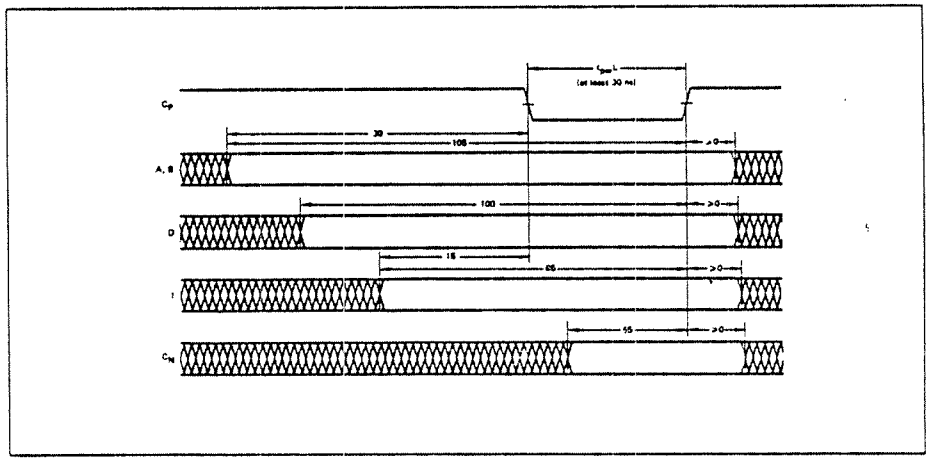
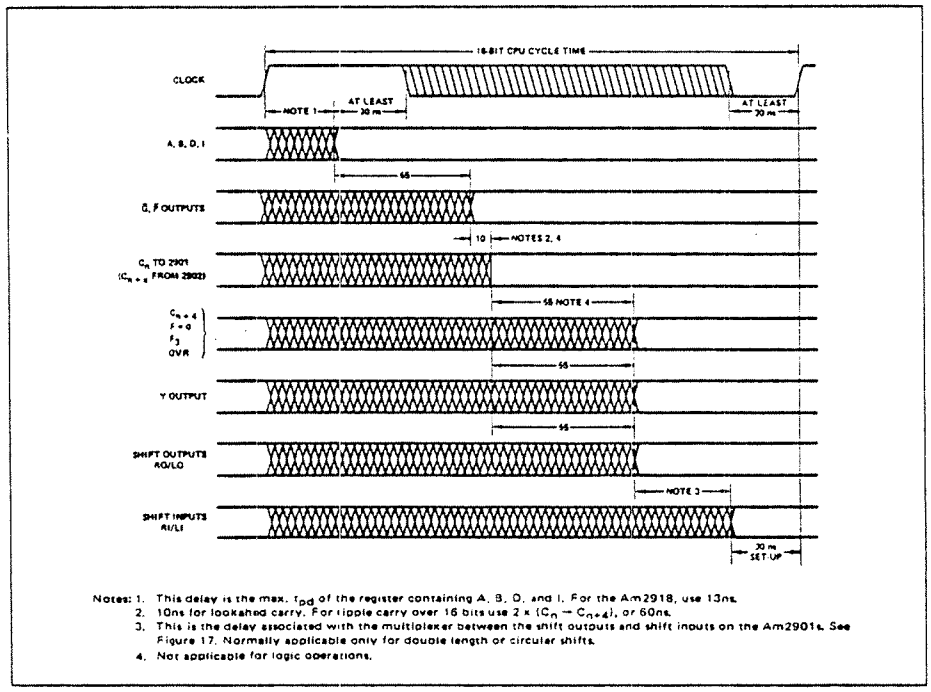


Figure 11. Minimum Cycle Times from Inputs. Numbers Shown are Minimum Data Stable Times for Am2901 DC, in ns.



Notes: 1. This delay is the max. t_{pd} of the register containing A, B, D, and I. For the Am2918, use 13ns.
2. 10ns for lookahed carry. For ripple carry over 16 bits use 2 x (C_n – C_{n+4}), or 60ns.
3. This is the delay associated with the multiplexer between the shift outputs and shift inputs on the Am2901s. See Figure 17. Normally applicable only for double length or circular shifts.
4. Not applicable for logic operations.

Figure 12. Switching Waveforms for 16-Bit System Assuming A, B, D and I are all Driven from Registers with the same Propagation Delay, Clocked by the Am2901 Clock.

APPENDIX C

DATA FOR AM 2914 INTERRUPT CONTROLLER

Taken from "AM2900 Bipolar Microprocessor Family" issued by Advanced Micro Devices, Inc. (to be contacted for more information).

Am2914

Vectored Priority Interrupt Controller

DISTINCTIVE CHARACTERISTICS

- Accepts 8 interrupt inputs
  Interrupts may be pulses or levels and are stored internally
- Built-in mask register
  Six different operations can be performed on mask register
- Built-in status register
  Status register holds code for lowest allowed interrupt
- Vectored output
  Output is binary code for highest priority un-masked interrupt
- Expandable
  Any number of Am2914's may be stacked for large interrupt systems
- Microprogrammable
  Executes 16 different microinstructions
  Instruction enable pin aids in vertical microprogramming
- High-speed operation
  Delay from an interrupt clocked into the interrupt register to interrupt request output is typically 60 ns

TABLE OF CONTENTS

Block Diagram 2-107
Connection Diagrams 2-108
Instructions 2-108
Ordering Information 2-109
DC Characteristics 2-110
AC Characteristics 2-111
Burn-in Circuit 2-113
Applications 2-114
Detailed Logic Description 2-130

FUNCTIONAL DESCRIPTION

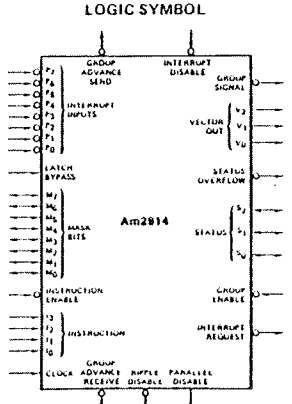
The Am2914 is a high-speed, eight-bit priority interrupt unit that is cascadable to handle any number of priority interrupt request levels. The high-speed of the Am2914 makes it ideal for use in Am2900 family microcomputer designs, but it can also be used with the Am9080A MOS microprocessor.

The Am2914 receives interrupt requests on 8 interrupt input lines (P_0-P_7). A LOW level is a request. An internal latch may be used to catch pulses on these lines, or the latch may be bypassed so the request lines drive the edge-triggered interrupt register directly. An 8-bit mask register is used to mask individual interrupts. Considerable flexibility is provided for controlling the mask register. Requests in the interrupt register are ANDed with the corresponding bits in the mask register and the results are sent to an 8-input priority encoder, which produces a three bit encoded vector representing the highest numbered input which is not masked.

An internal status register is used to point to the lowest priority at which an interrupt will be accepted. The contents of the status register are compared with the output of the priority encoder, and an interrupt request output will occur if the vector is greater than or equal to status. Whenever a vector is read from the Am2914 the status register is automatically updated to point to one level higher than the vector read. (The status register can be loaded externally or read out at any time using the S pins.) Signals are provided for moving the status upward across devices (Group Advance Send and Group Advance Receive) and for inhibiting lower priorities from higher order devices (Ripple Disable, Parallel Disable, and Interrupt Disable). A status overflow output indicates that an interrupt has been read at the highest priority.

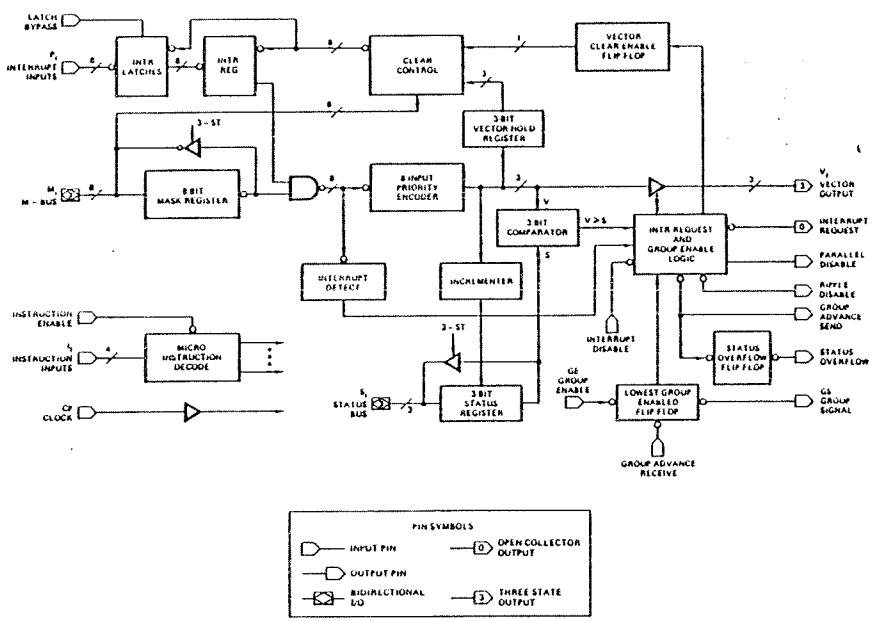
The Am2914 is controlled by a 4-bit instruction field I_0-I_3. The command on the instruction lines is executed if IE is LOW and is ignored if IE is HIGH, allowing the 4 I bits to be shared with other devices.

LOGIC SYMBOL



MPR-123

BLOCK DIAGRAM



MPR-124

BLOCK DIAGRAM DESCRIPTION

The Microinstruction Decode circuitry decodes the Interrupt Microinstructions and generates required control signals for the chip.

The Interrupt Register holds the Interrupt Inputs and is an eight-bit, edge-triggered register which is set on the rising edge of the CP Clock signal.

The Interrupt latches are set/reset-type latches. When the Latch Bypass signal is LOW, the latches are enabled and act as negative pulse catchers on the inputs to the Interrupt Register. When the Latch Bypass signal is HIGH, the Interrupt latches are transparent.

The Mask Register holds the eight mask bits associated with the eight interrupt levels. The register may be loaded from or read to the M Bus. Also, the entire register or individual mask bits may be set or cleared.

The Interrupt Detect circuitry detects the presence of any unmasked Interrupt Input. The eight-input Priority Encoder determines the highest priority, non-masked Interrupt Input and forms a binary coded interrupt vector. Following a Vector Read, the three-bit Vector Hold Register holds the binary coded interrupt vector. This stored vector is used for clearing interrupts.

The three-bit Status Register holds the status bits and may be loaded from or read to the S Bus. During a Vector Read, the Incrementer increments the interrupt vector by one, and the result is clocked into the Status Register. Thus the Status Register always points to the lowest level at which an interrupt will be accepted.

The three-bit Comparator compares the Interrupt Vector with the contents of the Status Register and indicates if the Interrupt Vector is greater than or equal to the contents of the Status Register.

The Lowest Group Enabled Flip-Flop is used when a number of 2914's are cascaded. In a cascaded system, only one Lowest Group Enabled Flip-Flop is LOW at a time. It indicates the eight interrupt group, which contains the lowest priority interrupt level which will be accepted and is used to form the higher order status bits.

The Interrupt Request and Group Enable logic contain various gating to generate the Interrupt Request, Parallel Disable, Ripple Disable, and Group Advance Send signals.

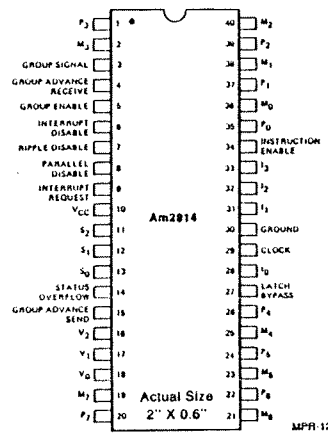
The Status Overflow signal is used to disable all interrupts. It indicates the highest priority interrupt vector has been read and the Status Register has overflowed.

The Clear Control logic generates the eight individual clear signals for the bits in the Interrupt Latches and Register. The Vector Clear Enable Flip-Flop indicates if the last vector read was from this group. When it is set, it enables the Clear Control Logic.

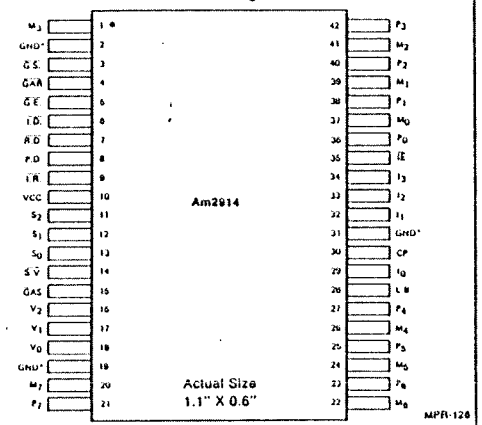
The CP clock signal is used to clock the Interrupt Register, Mask Register, Status Register, Vector Hold Register, and the Lowest Group Enabled, Vector Clear Enable and Status Overflow Flip-Flops, all on the clock LOW-to-HIGH transition.

CONNECTION DIAGRAMS – Top Views

DIP



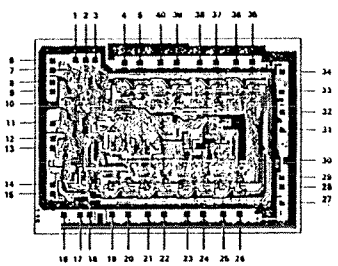
Flat Package



*Note: GND's and pins 2, 19, 31 must all be tied together externally.

Note: Pin 1 is marked for orientation.

Metallization and Pad Layout



DIE SIZE
133 mm X 187 mm

Numbers correspond to DIP pin-out.

TABLE I
MICROINSTRUCTION SET FOR Am2914 PRIORITY INTERRUPT CIRCUIT

Decimal $I_3I_2I_1I_0$	Mnemonic	Instruction
		Mask Register Functions
14	LDM	Load mask register from M bus
7	RDM	Read mask register to M bus
12	CLRM	Clear mask register (enables all priorities)
8	SETM	Set mask register (inhibits all interrupts)
10	BCLRM	Bit clear mask register from M bus
11	BSETM	Bit set mask register from M bus
		Status Register Functions
9	LDSTA	Load status register from S bus and LGE flip-flop from GE input
6	RDSTA	Read status register to S bus
		Interrupt Request Control
15	ENIN	Enable interrupt request
13	DISIN	Disable interrupt request
		Vectored Output
5	RDVC	Read vector output to V outputs, load V+1 into status register, load V into vector hold register and set vector clear enable flip-flop.
		Priority Interrupt Register Clear
1	CLRIN	Clear all interrupts
3	CLRMR	Clear interrupts from mask register data (uses the M bus)
2	CLRMB	Clear interrupts from M bus data
4	CLRVC	Clear the individual interrupt associated with the last vector read
		Master Clear
0	MCLR	Clear all interrupts, clear mask register, clear status register, clear LGE flip-flop, enable interrupt request.

STANDARD SCREENING
(Conforms to MIL-STD-883 for Class C Parts)

Step	MIL-STD-883 Method	Conditions	Level Am2914PC, DC	Level Am2914DM, FM
Pre-Seal Visual Inspection	2010	B	100%	100%
Stabilization Bake	1008	C 24-hour 150°C	100%	100%
Temperature Cycle	1010	C −65°C to +150°C 10 cycles	100%	100%
Centrifuge	2001	B 10,000 G	100% *	100%
Fine Leak	1014	A 5×10^{-8} atm-cc/sec	100% *	100%
Gross Leak	1014	C2 Fluorocarbon	100% *	100%
Electrical Test Subgroups 1 and 7	5004	See below for definitions of subgroups	100%	100%
Insert Additional Screening here for Class B Parts				
Group A Sample Tests	5005	See below for definitions of subgroups. Maximum accept number is 3		
Subgroup 1			LTPD = 5	LTPD = 5
Subgroup 2			LTPD = 7	LTPD = 7
Subgroup 3			LTPD = 7	LTPD = 7
Subgroup 7			LTPD = 7	LTPD = 7
Subgroup 8			LTPD = 7	LTPD = 7
Subgroup 9			LTPD = 7	LTPD = 7

*Not applicable for Am2914PC.

ORDERING INFORMATION

Order the part number according to the table below to obtain the desired package, temperature range, and screening level.

Order Number	Package Type (Note 1)	Temperature Range (Note 2)	Screening Level (Note 3)
AM2914PC	P-40	C	C-1
AM2914DC	D-40	C	C-1
AM2914DC-B	D-40	C	B-1
AM2914DM	D-40	M	C-3
AM2914DM-B	D-40	M	B-3
AM2914FM	F-42	M	C-3
AM2914FM-B	F-42	M	B-3
AM2914XC	Dice	C	Visual inspection to MIL-STD-883 Method 2010B.
AM2914XM	Dice	M	

Notes: 1. P = Molded DIP, D = Hermetic DIP, F = Flat Pak. Number following letter is number of leads. See Appendix B for detailed outline. Where Appendix B contains several dash numbers, any of the variations of the package may be used unless otherwise specified.
2. C = 0°C to +70°C, M = −55°C to +125°C.
3. See Appendix A for details of screening. Levels C-1 and C-3 conform to MIL-STD-883, Class C. Level B-3 conforms to MIL-STD-883, Class B.

MAXIMUM RATINGS (Above which the useful life may be impaired)

Storage Temperature	−65°C to +150°C
Temperature (Ambient) Under Bias	−55°C to +125°C
Supply Voltage to Ground Potential	−0.5V to +7.0V
DC Voltage Applied to Outputs for High Output State	+0.5V to +V_{CC} max.
DC Input Voltage	−0.5V to 5.5V
DC Output Current, Into Outputs	30mA
DC Input Current	−30mA to +5.0mA

OPERATING RANGE

P/N	Temperature	V_{CC}
Am2914PC, DC	0°C to +70°C	4.75V to 5.25V
Am2914DM, FM	−55°C to +125°C	4.50V to 5.50V

ELECTRICAL CHARACTERISTICS OVER OPERATING TEMPERATURE RANGE (Unless Otherwise Noted)

(Group A, Subgroups 1, 2, and 3)

Am2914XC T_A = 0°C to +70°C V_{CC} = 5.0V ± 5% (COM'L) MIN. = 4.75V MAX. = 5.25V

Am2914XM T_C = −55°C to +125°C V_{CC} = 5.0V ± 10% (MIL) MIN. = 4.50V MAX. = 5.50V

Parameters	Description	Test Conditions (Note 1)			Min.	Typ. (Note 2)	Max.	Units
V_{OH}	Output HIGH Voltage	V_{CC} = MIN., V_{IN} = V_{IH} or V_{IL}	MIL, I_{OH} = −1.0mA		2.4			Volts
			COM'L, I_{OH} = −2.6mA		2.4			
I_{CEX}	Output Leakage Current for IR Output	V_{CC} = MIN., V_O = 5.5V					250	µA
V_{OL}	Output LOW Voltage	V_{CC} = MIN., V_{IN} = V_{IH} or V_{IL}	I_{OL} = 4.0mA				0.4	Volts
			I_{OL} = 8.0mA				0.45	
			I_{OL} = 12mA				0.5	
V_{IH}	Input HIGH Level	Guaranteed input logical HIGH voltage for all inputs			2.0			Volts
V_{IL}	Input LOW Level	Guaranteed input logical LOW voltage for all inputs					0.8	Volts
V_I	Input Clamp Voltage	V_{CC} = MIN., I_{IN} = −18mA					−1.5	Volts
I_{IL}	Input LOW Current	V_{CC} = MAX., V_{IN} = 0.4V	M_{0-7}				−0.15	mA
			S_{0-2}				−0.1	
			L. B.				−0.4	
			$\overline{I.D.}$				−2.0	
			All Others				−0.8	
I_{IH}	Input HIGH Current	V_{CC} = MAX., V_{IN} = 5.5V	$\overline{IE}$, $\overline{GE}$, $\overline{GAR}$, S_{0-2}				40	µA
			M_{0-7}, $\overline{I.D.}$				60	
			All Others				20	
I_I	Input HIGH Current	V_{CC} = MAX., V_{IN} = 5.5V					1.0	mA
I_O	Off-State Output Current	V_{CC} = MAX.	V_{OUT} = 0.5V	M_{0-7}			−150	µA
				S_{0-2}			−100	
				V_{0-2}			−50	
			V_{OUT} = 2.4V	M_{0-7}			150	
				S_{0-2}			100	
				V_{0-2}			50	
I_{CC}	Power Supply Current	V_{CC} = 5.0V, 25°C				170	–	mA
		V_{CC} = MAX.	COM'L	0°C			305	
				70°C			250	
			MIL	−55°C			310	
				125°C			200	
I_{SC}	Output Short Circuit Current (Note 3)	V_{CC} = MAX.			−40		−100	mA

Notes: 1. For conditions shown as MIN. or MAX., use the appropriate value specified under Electrical Characteristics for the applicable device type.
2. Typical limits are at V_{CC} = 5.0V, 25°C ambient and maximum loading.
3. Not more than one output should be shorted at a time. Duration of the short circuit test should not exceed one second.

SWITCHING CHARACTERISTICS AT 25°C AND 5.0 VOLTS

Note: Guaranteed limits at 25°C and 5.0V are group A, subgroup 9 tests
All outputs fully loaded, C_L = 50pF. Measurements made at 1.5V with
input levels of 0V and 3.0V. All numbers are in ns.
For interrupt request output, R_L = 470Ω

TABLE I. CLOCK AND INTERRUPT INPUT PULSE WIDTHS (ns)

Time	GUARANTEED
Minimum Clock LOW Time	30
Minimum Clock HIGH Time	30
Minimum Interrupt Input (P_0-P_7) LOW Time for Guaranteed Acceptance (Pulse Mode)	25
Maximum Interrupt Input (P_0-P_7) LOW Time for Guaranteed Rejection (Pulse Mode)	10

TABLE II. COMBINATIONAL PROPAGATION DELAYS (ns)

	TYPICAL						GUARANTEED					
From Input \ To Output	M Bus	S Bus	V_{012}	Irpt Req	Ripple Disable	Group Advance Send	M Bus	S Bus	V_{012}	Irpt Req	Ripple Disable	Group Advance Send
$\overline{IE}$	36	40	40	–	–	30	48	65	65	–	–	47
I_{0123}	36	40	40	–	–	30	48	65	65	–	–	47
Irpt. Disable	–	–	25	35	8	19	–	–	37	42	13	23

R_L = 2.0kΩ, C_L = 15pF

TABLE III. DELAYS FROM CLOCK TO OUTPUTS (ns)

Clock Path	TYPICAL							GUARANTEED						
	To V_{012}	To Irpt Req	To PD	To $\overline{RD}$	To $\overline{GAS}$	To Status O'flow	To $\overline{GS}$	To V_{012}	To Irpt Req	To PD	To $\overline{RD}$	To $\overline{GAS}$	To Status O'flow	To $\overline{GS}$
Irpt Latches and Register	55	65	37	39	47	–	–	67	82	57	57	66	–	–
Mask Register	55	65	37	39	47	–	–	67	82	57	57	66	–	–
Status Register	45	55	28	31	37	–	–	59	74	57	57	58	–	–
Lowest Group Enabled Flip Flop	–	–	22	25	–	–	17	–	–	42	45	–	–	32
Irpt Request Enable Flip-Flop	–	40	–	–	–	–	–	–	68	–	–	–	–	–
Status Overflow Flip-Flop	–	–	–	–	–	17	–	–	–	–	–	–	26	–

TABLE IV. SET-UP AND HOLD TIME REQUIREMENTS (ns)

(All relative to clock LOW-to-HIGH transition)

From Input	GUARANTEED	
	Set-up Time	Hold Time
S-Bus	11	8
M-Bus	11	8
$\overline{P_0}$-$\overline{P_7}$	11	6
Latch Bypass	16	0
$\overline{IE}$ I_{0123} (See Note)	46 t_{pwL} + 29	0
$\overline{GE}$	11	11
$\overline{GAR}$	11	11
Irpt Disable	35	0
P-...-P_7 Hold Time tive to LB	–	16

Note: t_{pwL} is the Clock LOW Time. Both Set-up times must be met.

SWITCHING CHARACTERISTICS OVER OPERATING VOLTAGE AND TEMPERATURE RANGE

(Group A, subgroup 10 and 11 tests and limits)

All outputs fully loaded, C_L = 50pF. Measurements made at 1.5V with input levels of 0V and 3.0V. For Interrupt Request Output, R_L = 470Ω.

TABLE V. CLOCK AND INTERRUPT INPUT PULSE WIDTHS (ns)

Time	Am2914PC, DC, XC T_A = 0°C to +70°C, 5V ± 5%	Am2914DM, FM, XM T_C = −55°C to +125°C, 5V ± 10%
Minimum Clock LOW Time	30	30
Minimum Clock HIGH Time	30	30
Minimum Interrupt Input ($\overline{P_0}$-$\overline{P_7}$) LOW Time for Guaranteed Acceptance (Pulse Mode)	40	40
Maximum Interrupt Input ($\overline{P_0}$-$\overline{P_7}$) LOW Time for Guaranteed Rejection (Pulse Mode)	8	8
Minimum Clock Period, $\overline{IE}$ = H on current cycle and previous cycle	60	55
Minimum Clock Period, $\overline{IE}$ = L on current cycle or previous cycle	100	110

TABLE VI. MAXIMUM COMBINATIONAL PROPAGATION DELAYS (ns)

	Am2914PC, DC, XC T_A = 0°C to +70°C, 5V ± 5%						Am2914DM, FM, XM T_C = −55°C to +125°C, 5V ± 10%					
To Output / From Input	M Bus	S Bus	V_{012}	Irpt Req	Ripple Disable	Group Advance Send	M Bus	S Bus	V_{012}	Irpt Req	Ripple Disable	Group Advance Send
$\overline{IE}$	52	60	60	–	–	56	60	68	68	–	–	62
I_{0123}	52	60	60	–	–	56	60	68	68	–	–	62
Irpt. Disable	–	–	40	52	14	27	–	–	46	60	15	33

R_L = 2.0kΩ, C_L = 15pF

TABLE VII. MAXIMUM DELAYS FROM CLOCK TO OUTPUTS (ns)

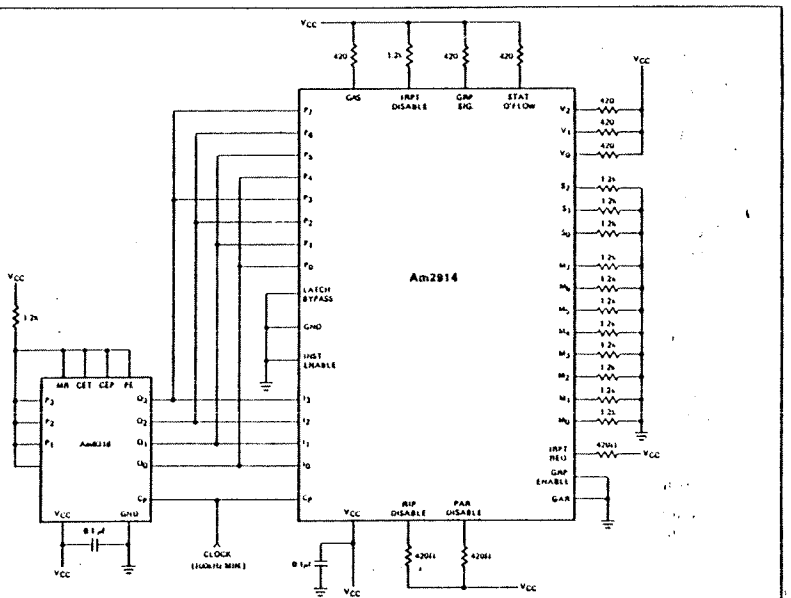
Clock Path	Am2914PC, DC, XC T_A = 0°C to +70°C, 5V ± 5%							Am2914DM, FM, XM T_C = −55°C to +125°C, 5V ± 10%						
	To V_{012}	To Irpt Req	To PD	To $\overline{RD}$	To $\overline{GAS}$	To Status O'flow	To $\overline{GS}$	To V_{012}	To Irpt Req	To PD	To $\overline{RD}$	To $\overline{GAS}$	To Status O'flow	To $\overline{GS}$
Irpt Latches and Register	76	97	67	67	80	–	–	82	105	75	75	85	–	–
Mask Register	76	97	67	67	80	–	–	82	105	75	75	85	–	–
Status Register	67	88	63	63	70	–	–	73	96	66	66	76	–	–
Lowest Group Enabled Flip Flop	–	–	48	52	–	–	38	–	–	54	58	–	–	45
Irpt Request Enable Flip Flop	–	62	–	–	–	–	–	–	66	–	–	–	–	–
Status Overflow Flip Flop	–	–	–	–	–	30	–	–	–	–	–	–	35	–

TABLE VIII. SET-UP AND HOLD TIME REQUIREMENTS (ns)

(All relative to clock LOW-to-HIGH transition)

From Input	Am2914PC, DC, XC T_A = 0°C to +70°C, 5V ± 5%		Am2914DM, FM, XM T_C = −55°C to +125°C, 5V ± 10%	
	Set-Up Time	Hold Time	Set-Up Time	Hold Time
S Bus	15	10	15	10
M Bus	15	10	15	10
$\overline{P_0}$-$\overline{P_7}$	15	8	15	8
Latch Bypass	20	0	20	0
$\overline{IE}$ I_{0123} (See Note)	55 t_{pwL} + 33	0	55 t_{pwL} + 40	0
$\overline{GE}$	15	13	15	13
$\overline{GAR}$	15	13	15	13
Irpt Disable	42	0	42	0
$\overline{P_0}$-$\overline{P_7}$ Hold Time Relative to LB	—	20	—	20

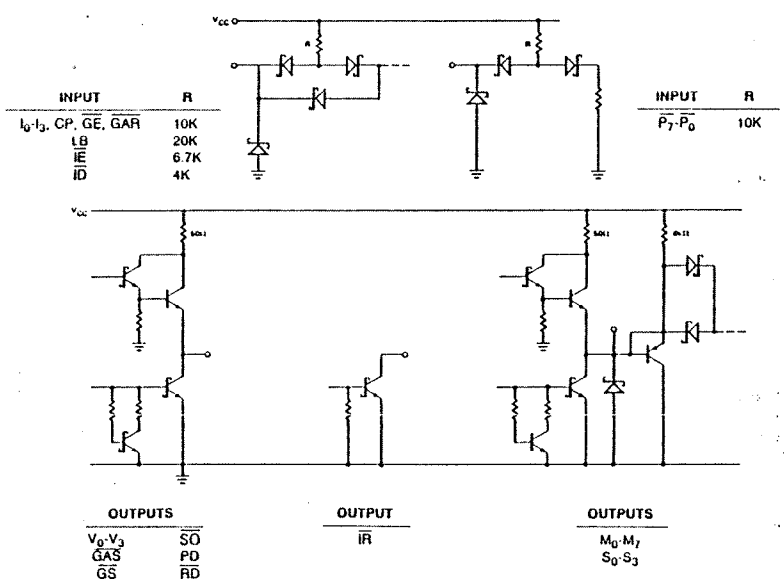
Note: t_{pwL} is the Clock LOW Time. Both Set-up times must be met.



Am2914 Burn-in Circuit

MPR-127

INPUT/OUTPUT CIRCUITS



MPR-128

A MICROPROGRAMMABLE, BIPOLAR, LSI INTERRUPT STRUCTURE USING THE Am2914

INTRODUCTION

Advanced Micro Devices' introduction of the Am2914 Vectored Priority Interrupt Controller now makes possible the structuring of a microprogrammable bipolar LSI interrupt system. The design engineer may use the Am2914 to simplify his design process, dramatically reduce the system cost, size and package count, and increase the speed, capability and reliability of his interrupt system.

The Am2914 is a modular, low cost, standard LSI component that may be microprogrammed to meet the requirements of specific applications. Today's engineer may utilize the Am2914 microprogrammability to provide functional flexibility and ease of engineering change, while taking advantage of its modularity to provide hardware regularity and future expansion capability.

THE INTERRUPT CONCEPT

In any state machine, a requirement exists for the efficient synchronization and response to asynchronous events such as power failure, machine malfunctions, control panel service requests, external timer signals, supervisory calls, program errors, and input/output device service requests. The merit of such an "asynchronous event handler" may be measured in terms of response time, system throughput, real time overhead, hardware cost and memory space required.

The simplest approach to asynchronous event handling is the poll approach. A status indicator is associated with each possible asynchronous event. The processor tests each indicator in sequence and, in effect, "asks" if service is required. This program-driven method is inefficient for a number of reasons. Much time is consumed polling when no service is required; programs must have frequent test points to poll indicators, and since indicators are polled in sequence, considerable time may elapse before the processor responds to an event. Thus, system throughput is low; real time overhead and response time are high, and a large memory space is required.

The interrupt method is a much more efficient way of servicing asynchronous requests. An asynchronous event requiring service generates an interrupt request signal to the processor. When the processor receives the interrupt request, it may suspend the program it is currently executing, execute an interrupt service routine which services the asynchronous request, then resume the execution of the suspended program. In this system, the execution of the service routine is initiated by an interrupt request; thus, the system is interrupt driven and service routines are executed only when service is requested. Although hardware cost may be higher in this type of system, it is more efficient since system throughput is higher, response time is faster, real time overhead is lower and less memory space is required.

INTERRUPT SYSTEM FUNCTIONAL DEFINITION

A complete and clear functional definition is key to the design of a good interrupt system. The following features are useful.

Multiple Interrupt Request Handling: Since interrupt requests are generated from a number of different sources, the interrupt system's ability to handle interrupt requests from several sources is important.

Interrupt Request Prioritization: Since the processor can service only one interrupt request at a time, it is important that the interrupt system has the ability to prioritize the requests and determine which has the highest priority.

Interrupt Service Routine "Nesting": This feature allows an interrupt service routine for a given priority request to be interrupted in turn, but only by a higher priority interrupt request. The service routine for the higher priority request is executed, then the execution of the interrupted service routine is resumed. If there are "n" interrupt requests, an "n" deep "nest" is possible.

Dynamic Interrupt Enabling/Disabling: The ability to enable/disable all interrupts "on the fly" under microprogram control can be used to prevent interruption of certain processes.

Dynamic Interrupt Request Masking: The ability to selectively inhibit or "mask" individual interrupt requests under microprogram control is useful.

Interrupt Request Vectoring: Many times, a particular interrupt request requires the execution of a unique interrupt service routine. For this reason, the generation of a unique binary coded vector for each interrupt request is very helpful. This vector can be used as a pointer to the start of a unique service routine.

Interrupt Request Priority Threshold: The ability to establish a priority threshold is valuable. In this type of operation, only those interrupt requests which have higher priority than a specified threshold priority are accepted. The threshold priority can be defined by microprogram or can be automatically established by hardware at the interrupt currently being serviced plus one. This automatic threshold prevents multiple interrupts from the same source. Also useful is the ability to read the threshold priority under microprogram control. Thus, the interrupt request being serviced may be determined by the microprogram.

Interrupt Request Clearing Flexibility: Flexibility in the method of clearing interrupt requests allows different modes of interrupt system operation. Of particular value are the abilities to clear the interrupt currently being serviced, clear all interrupts, or clear interrupts via a programmable mask register or bus.

Microprogrammability: Microprogrammability permits the construction of a general purpose or "universal" interrupt structure which can be microprogrammed to meet a specific application's requirements. The universality of the structure allows standardization of the hardware and amortization of the hardware development costs across a much broader user base. The end result is a flexible, low cost interrupt structure.

Hardware Modularity: Modular interrupt system hardware is beneficial in two ways. First, hardware modularity provides expansion capability. Additional modules may be added as the need to service additional requests arises. Secondly, hardware modularity provides a structural regularity which simplifies the system structure and also reduces the number of hardware part numbers.

Fast Interrupt System Response Time: Quick interrupt system response provides more efficient system operation. Fast response reduces real time overhead and increases overall system throughput.

INTERRUPT SYSTEM IMPLEMENTATION USING THE Am2914

The Am2914 provides all of the foregoing features on a single LSI chip. The Am2914 is a high-speed, eight-bit priority interrupt unit that is cascadable to handle any number of priority interrupt request levels. The Am2914's high speed is ideal for use in Am2900 Family microcomputer designs, but it can also be used with the Am9080A MOS microprocessor.

The Am2914 receives interrupt requests on eight interrupt input lines (P_0-P_7). A LOW level is a request. An internal latch may be used to catch pulses (HIGH-LOW-HIGH) on these lines, or the latch may be bypassed so that the request lines drive the D-inputs to the edge-triggered Interrupt Register directly. An eight-bit Mask Register is used to mask individual interrupts. Considerable flexibility is provided for controlling the Mask Register. Requests in the Interrupt Register (P_0-P_7) are ANDed with the corresponding bits in the mask register (M_0-M_7) and the results are sent to an eight-input priority encoder, which produces a three-bit encoded vector representing the highest priority input which is not masked.

An internal Status Register is used to point to the lowest priority at which an interrupt will be accepted. The contents of the Status Register are compared with the output of the priority encoder, and an Interrupt Request output will occur if the vector is greater than or equal to the contents of the Status Register. Whenever a vector is read from the Am2914, the Status Register is automatically updated to point to one level higher than the vector read. (The Status Register can be loaded externally or read out at any time using the S-Bus.) Signals are provided for moving the status upward across devices (Group Advance Send and Group Advance Receive) and for inhibiting lower priorities from higher order devices (Ripple Disable, Parallel Disable, and Interrupt Disable). A Status Overflow output indicates that an interrupt has been read at the highest priority.

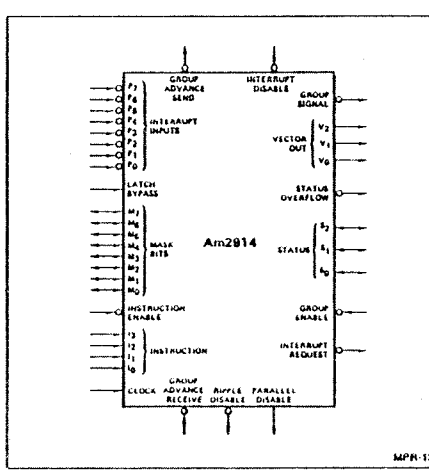


Figure 1. Am2914 Logic Symbol.

The Am2914 is controlled by a four-bit microinstruction field I_0-I_3. The microinstruction is executed if $\overline{IE}$ (Instruction Enable) is LOW and is ignored if $\overline{IE}$ is HIGH, allowing the four I bits to be shared with other functions. Sixteen different microinstructions are executed. Figure 2 shows the microinstructions and the microinstruction codes.

MICROINSTRUCTION DESCRIPTION	MICROINSTRUCTION CODE $I_3I_2I_1I_0$
MASTER CLEAR	0000
CLEAR ALL INTERRUPTS	0001
CLEAR INTERRUPTS FROM M-BUS	0010
CLEAR INTERRUPTS FROM MASK REGISTER	0011
CLEAR INTERRUPT, LAST VECTOR READ	0100
READ VECTOR	0101
READ STATUS REGISTER	0110
READ MASK REGISTER	0111
SET MASK REGISTER	1000
LOAD STATUS REGISTER	1001
BIT CLEAR MASK REGISTER	1010
BIT SET MASK REGISTER	1011
CLEAR MASK REGISTER	1100
DISABLE INTERRUPT REQUEST	1101
LOAD MASK REGISTER	1110
ENABLE INTERRUPT REQUEST	1111

Figure 2. Am2914 Microinstruction Set.

In this microinstruction set, the *Master Clear* microinstruction is selected as binary zero so that during a power up sequence, the microinstruction register in the microprogram control unit of the central processor can be cleared to all zeros. Thus, on the next clock cycle, the Am2914 will execute the *Master Clear* function. This includes clearing the Interrupt Latches and Register as well as the Mask Register and Status Register. The LGE flip-flop of the least significant group is set LOW because the Group Advance Receive input is tied LOW. All other Group Advance Receive inputs are tied to Group Advance Send outputs and these are forced HIGH during this instruction. This clear instruction also sets the Interrupt Request Enable flip-flop so that a fully interrupt driven system can be easily initiated from any interrupt.

The *Clear All Interrupts* microinstruction clears the Interrupt Latches and Register.

The *Clear Interrupts from Mask Register* microinstruction clears those Interrupt Latches and Register bits which have corresponding Mask Register bits set equal to one. The M-Bus is used by the Am2914 during the execution of this microinstruction and must be floating.

The *Clear Interrupts from M-Bus* microinstruction clears those Interrupt Latches and Register bits which have corresponding M-Bus bits set equal to one.

The *Clear Interrupt, Last Vector Read* microinstruction clears the Interrupt Latch and Register bit associated with the last vector read.

The *Read Vector* microinstruction is used to read the vector value of the highest priority request causing the interrupt. The vector outputs are three-state drivers that are enabled onto the $V_0V_1V_2$ bus during this instruction. This microinstruction also automatically loads the value "vector plus one" into the Status Register. In addition, this instruction sets the Vector Clear Enable flip-flop and loads the current vector value into the Vector Hold Register so that this value can be used by the *Clear Interrupt, Last Vector Read* microinstruction. This allows the user to read the vector associated with the interrupt, and at some later time clear the Interrupt Latch and Register bit associated with the vector read.

The *Load Status Register* microinstruction loads S-Bus data into the Status Register and also loads the LGE flip-flop from the Group Enable input.

During the *Read Status Register* microinstruction, the Status Register outputs are enabled onto the Status Bus (S_0-S_2). The Status Bus is a three-bit, bi-directional, three-state bus.

The *Load Mask Register* microinstruction loads data from the three-state, bi-directional M-Bus into the Mask Register.

The *Read Mask Register* microinstruction enables the Mask Register outputs onto the bi-directional, three-state M-Bus.

The *Set Mask Register* microinstruction sets all the bits in the Mask Register to one. This results in all interrupts being inhibited.

The entire Mask Register is cleared by the *Clear Mask Register* microinstruction. This enables all interrupts subject to the Interrupt Enable flip-flop and the Status Register.

The *Bit Clear Mask Register* microinstruction may be used to selectively clear individual Mask Register bits. This microinstruction clears those Mask Register bits which have corresponding M-Bus bits equal to one. Mask Register bits with corresponding M-Bus bits equal to zero are not affected.

The *Bit Set Mask Register* microinstruction sets those Mask Register bits which have corresponding M-Bus bits equal to one. Other Mask Register bits are not affected.

All Interrupt Requests may be disabled by execution of the *Disable Interrupt Request* microinstruction. This microinstruction resets an Interrupt Request Enable flip-flop on the chip.

The *Enable Interrupt Request* microinstruction sets the Interrupt Enable flip-flop. Thus, Interrupt Requests are enabled subject to the contents of the Mask and Status Registers.

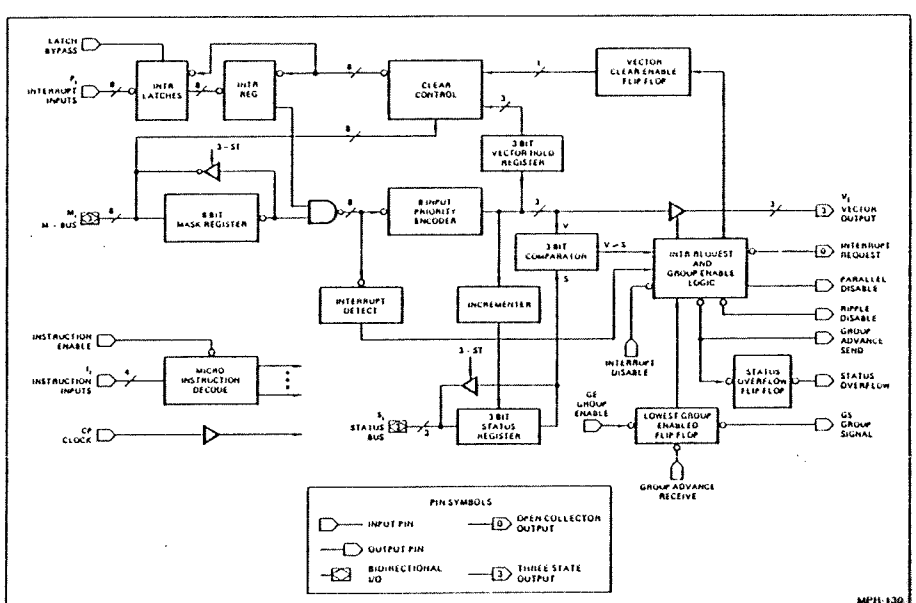


Figure 3. Am2914 Block Diagram.

Am2914 BLOCK DIAGRAM DESCRIPTION

The Am2914 block diagram is shown in Figure 3. The Micro-instruction Decode circuitry decodes the Interrupt Micro-instructions and generates required control signals for the chip.

The Interrupt Register holds the Interrupt Inputs and is an eight-bit, edge-triggered register which is set on the rising edge of the CP Clock signal if the Interrupt Input is LOW.

The Interrupt latches are set/reset latches. When the Latch Bypass signal is LOW, the latches are enabled and act as negative pulse catchers on the inputs to the Interrupt Register. When the Latch Bypass signal is HIGH, the Interrupt latches are transparent.

The Mask Register holds the eight mask bits associated with the eight interrupt levels. The register may be loaded from or read to the M-Bus. Also, the entire register or individual mask bits may be set or cleared.

The Interrupt Detect circuitry detects the presence of any unmasked Interrupt Input. The eight-input Priority Encoder determines the highest priority, non-masked Interrupt Input and forms a binary coded interrupt vector. Following a Vector Read, the three-bit Vector Hold Register holds the binary coded interrupt vector. This stored vector can be used later for clearing interrupts.

The three-bit Status Register holds the status bits and may be loaded from or read to the S-Bus. During a Vector Read, the Incrementer increments the interrupt vector by one, and the result is clocked into the Status Register. Thus, the Status Register points to a level one greater than the vector just read.

The three-bit Comparator compares the Interrupt Vector with the contents of the Status Register and indicates if the Interrupt Vector is greater than or equal to the contents of the Status Register.

The Lowest Group Enabled Flip-Flop is used when a number of Am2914's are cascaded. In a cascaded system, only one Lowest Group Enabled Flip-Flop is LOW at a time. It indicates the eight interrupt group, which contains the lowest priority interrupt level which will be accepted and is used to form the higher order status bits.

The Interrupt Request and Group Enable logic contain various gating to generate the Interrupt Request, Parallel Disable, Ripple Disable, and Group Advance Send signals.

The Status Overflow signal is used to disable all interrupts. It indicates the highest priority interrupt vector has been read and the Status Register has overflowed.

The Clear Control logic generates the eight individual clear signals for the bits in the Interrupt Latches and Register. The Vector Clear Enable Flip-Flop indicates if the last vector read was from this chip. When it is set it enables the Clear Control Logic.

The CP clock signal is used to clock the Interrupt Register, Mask Register, Status Register, Vector Hold Register, and the Lowest Group Enabled, Vector Clear Enable and Status Overflow Flip-Flops, all on the clock LOW-to-HIGH transition.

The Am2914 can be microprogrammed in many different ways. Figure 4 shows an example interrupt sequence. The *Read Vector* microinstruction is necessary in order to read the interrupt priority level. Since vector plus one is automatically loaded into the Status Register when a *Read Vector* microinstruction is executed, the Status Register possibly will overflow and disable all interrupts. For this reason, the Status Register must be reloaded periodically. The other Am2914 microinstructions are optional.

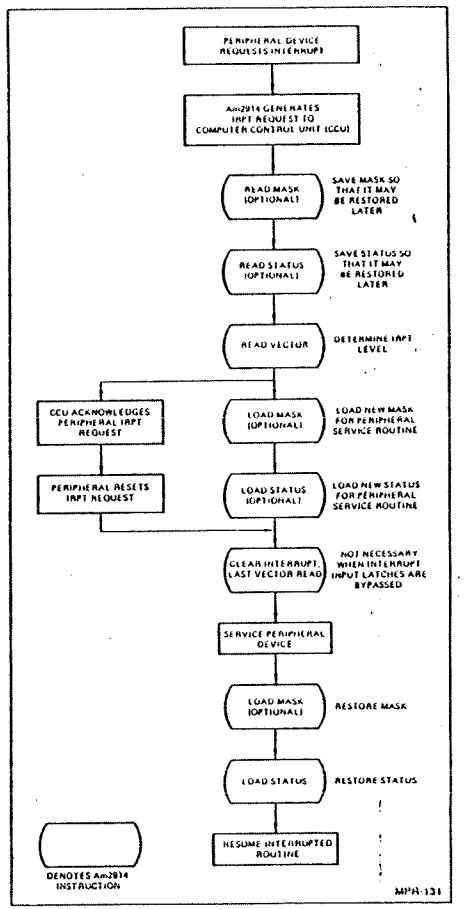


Figure 4. Example Interrupt Sequence.

CASCADING THE Am2914

A number of input/output signals are provided for cascading the Am2914 Vectored Priority Interrupt Encoder. A definition of these I/O signals and their required connections follows:

Group Signal ($\overline{GS}$) – This signal is the output of the Lowest Group Enabled flip-flop and during a Read Status micro-instruction is used to generate the high order bits of the Status word.

Group Enable ($\overline{GE}$) – This signal is one of the inputs to the Lowest Group Enable flip-flop and is used to load the flip-flop during the Load Status microinstruction.

Group Advance Send ($\overline{GAS}$) – During a Read Vector microinstruction, this output signal is LOW when the highest priority vector (vector seven) of the group is being read. In a cascaded system Group Advance Send must be tied to the Group Advance Receive input of the next higher group in order to transfer status information.

Group Advance Receive ($\overline{GAR}$) – During a Master Clear or Read Vector microinstruction, this input signal is used with other internal signals to load the Lowest Group Enabled flip-flop. The Group Advance Receive input of the lowest priority group must be tied to ground.

Status Overflow ($\overline{SV}$) – This output signal becomes LOW after the highest priority vector (vector seven) of the group has been read and indicates the Status Register has overflowed. It stays LOW until a Master Clear or Load Status microinstruction is executed. The Status Overflow output of the highest priority group should be connected to the Interrupt Disable input of the same group and serves to disable all interrupts until new status is loaded or the system is master cleared. The Status Overflow outputs of lower priority groups should be left open (see Figure 7).

Interrupt Disable ($\overline{ID}$) – When LOW, this input signal inhibits the Interrupt Request output from the chip and also generates a Ripple Disable output.

Ripple Disable ($\overline{RD}$) – This output signal is used only in the Ripple Cascade Mode (see below). The Ripple Disable output is LOW when the Interrupt Disable input is LOW, the Lowest Group Enabled flip-flop is LOW, or an Interrupt Request is generated in the group. In the ripple cascade mode, the

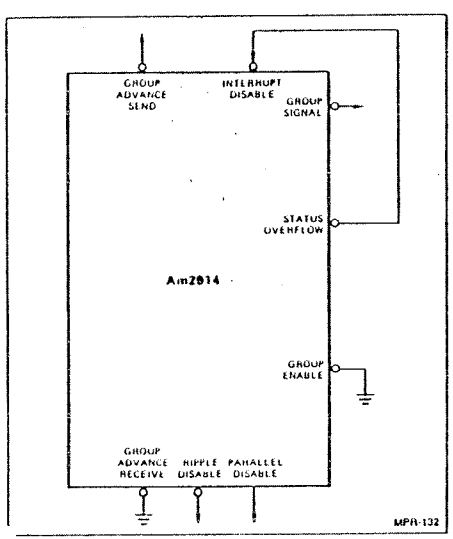


Figure 5. Cascade Lines Connection for Single Chip System.

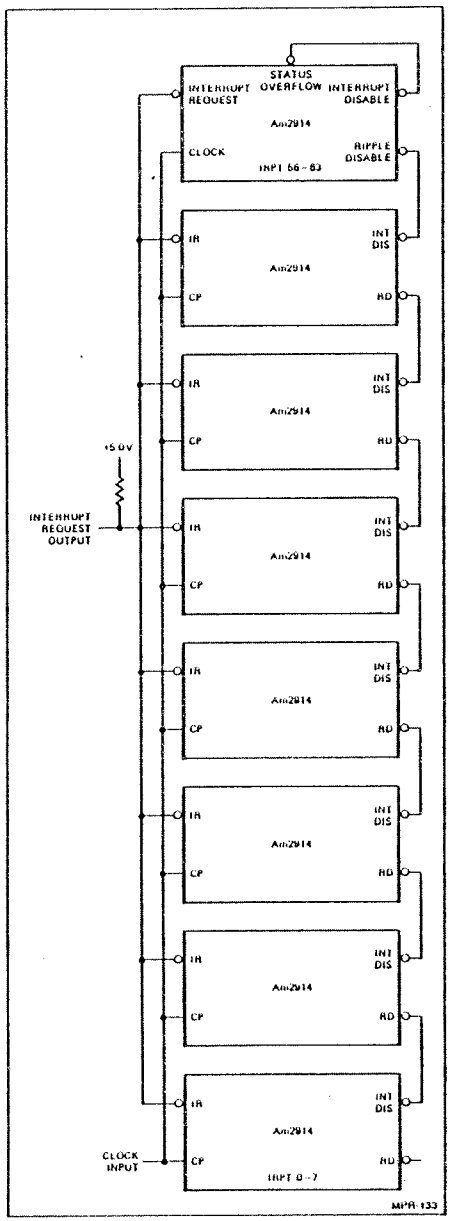


Figure 6. Interrupt Disable Connections for Ripple Cascade Mode.

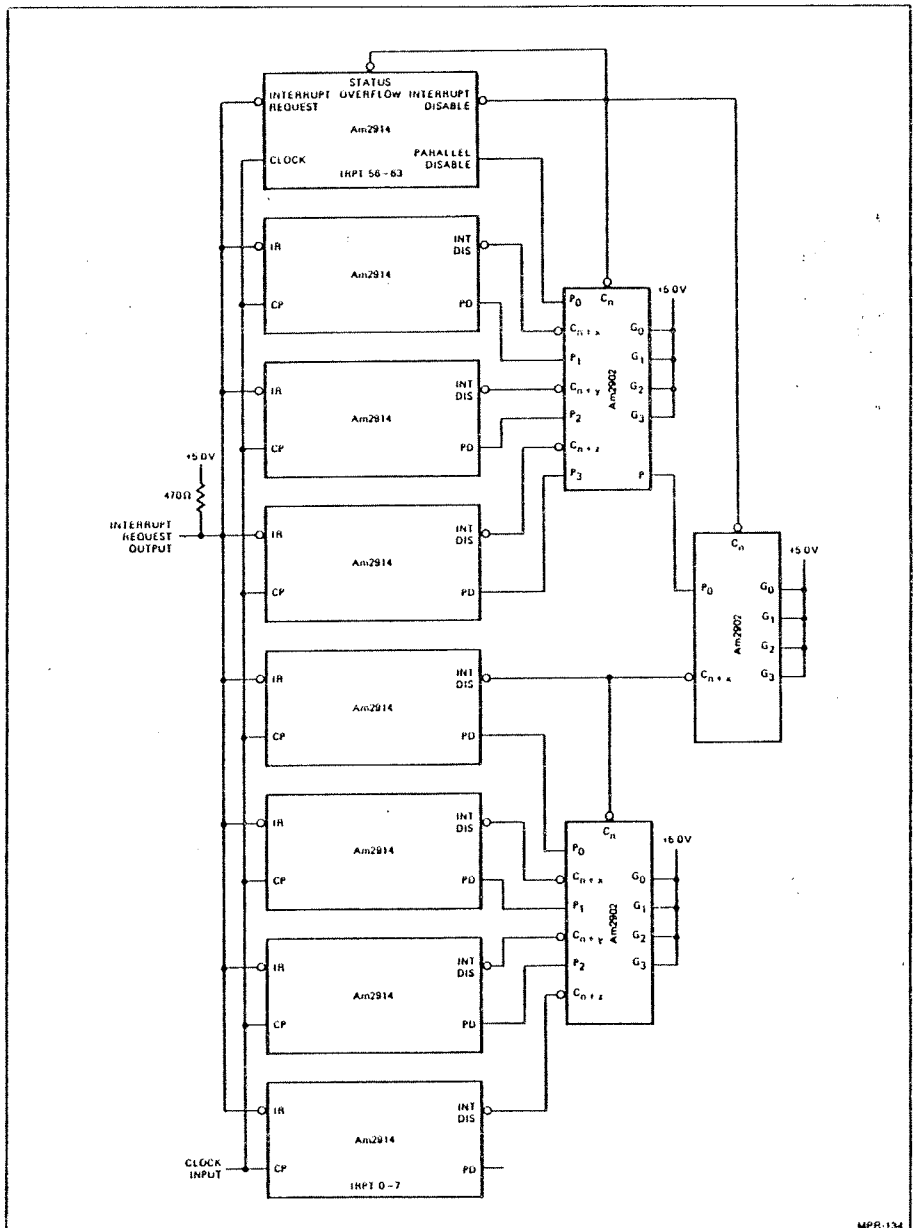


Figure 7. Interrupt Disable Connections for Parallel Cascade Mode.

Ripple Disable output is tied to the Interrupt Disable input of the next lower priority group (see Figure 6).

Parallel Disable (PD) – This output is used only in the parallel cascade mode (see below). It is HIGH when the Lowest Group Enabled flip-flop is LOW or an Interrupt Request is generated in the group. It is not affected by the Interrupt Disable input.

A single Am2914 chip may be used to prioritize and encode up to eight interrupt inputs. Figure 5 shows how the above cascade lines should be connected in such a single chip system.

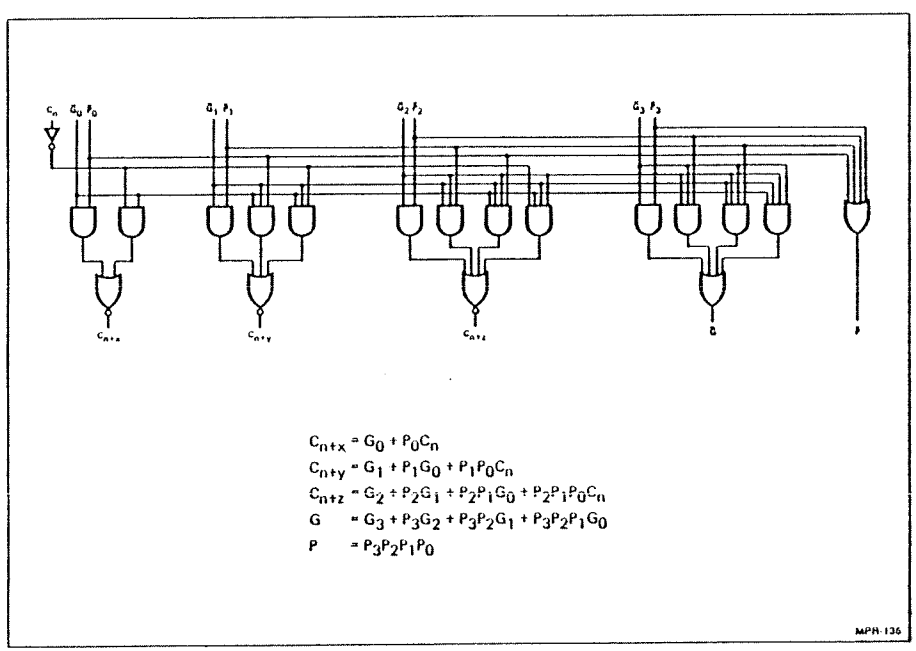
The Group Advance Receive and Group Enable inputs should be connected to ground so that the Lowest Group Enabled flip-flop is forced LOW during a *Master Clear* or *Load Status* microinstruction. Status Overflow should be connected to Interrupt Disable in order to disable interrupts when vector seven is read. The Group Advance Send, Ripple Disable, Group Signal and Parallel Disable pins should be left open.

The Am2914 may be cascaded in either a Ripple Cascade Mode or a Parallel Cascade Mode. In the Ripple Cascade Mode, the Interrupt Disable signal, which disables lower priority interrupts, is allowed to ripple through lower priority groups. Figures 6, 9 and 11 show the cascade connections required for a ripple cascade 64 input interrupt system.

In the parallel cascade mode, a parallel lookahead scheme is employed using the high-speed Am2902 Lookahead Carry Generator. Figures 7, 9 and 10 show the cascade connections required for a parallel cascade 64-input interrupt system. For this application, the Am2902 is used as a lookahead interrupt disable generator. A Parallel Disable output from any group results in the disabling of all lower priority groups in parallel. Figure 8 shows the Am2902 logic diagram and equations.

In Figures 9 and 10, the Am2913 Priority Interrupt Expander is shown forming the high order bits of the vector and status, respectively. The Am2913 is an eight-line to three-line priority encoder with three-state outputs which are enabled by the five output control signals G1, G2, $\overline{G3}$, $\overline{G4}$, and $\overline{G5}$. In Figure 9, the Am2913 is connected so that its outputs are enabled during a Read Vector instruction, and in Figure 10 the Am2913 is connected so that its outputs are enabled during a Read Status instruction. The Am2913 logic diagram and truth table are shown in Figure 11.

The Am25LS138 three-line to eight-line Decoder also is shown in Figure 10. It is used to decode the three high order status bits during a Load Status instruction. The Am25LS138 logic diagram and truth table are shown in Figure 12.



$$C_{n+x} = G_0 + P_0C_n$$
$$C_{n+y} = G_1 + P_1G_0 + P_1P_0C_n$$
$$C_{n+z} = G_2 + P_2G_1 + P_2P_1G_0 + P_2P_1P_0C_n$$
$$G = G_3 + P_3G_2 + P_3P_2G_1 + P_3P_2P_1G_0$$
$$P = P_3P_2P_1P_0$$

Figure 8. Am2902 Carry Look-Ahead Generator Logic Diagram and Equations.

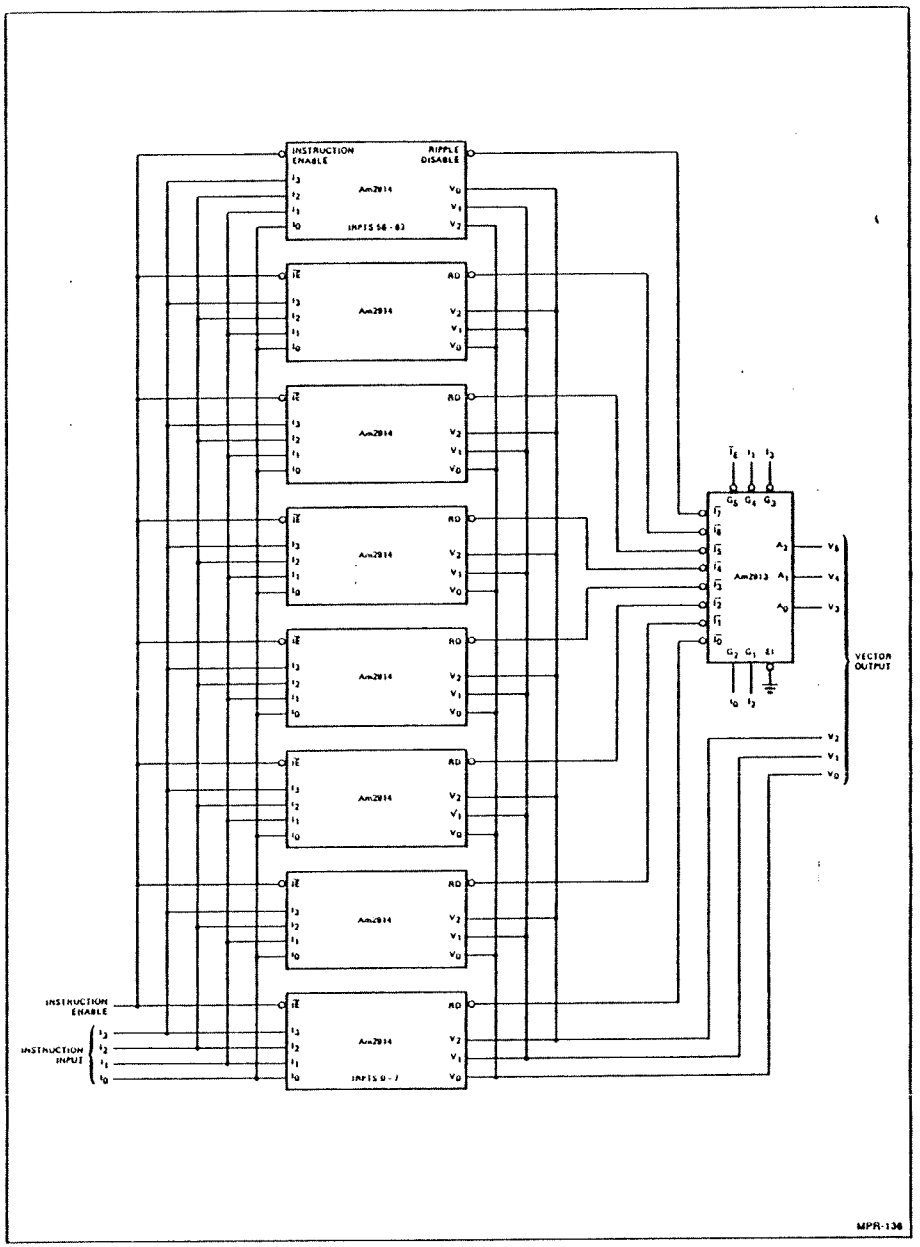


Figure 9. Vector Connections for both the Parallel and Ripple Cascade Modes.

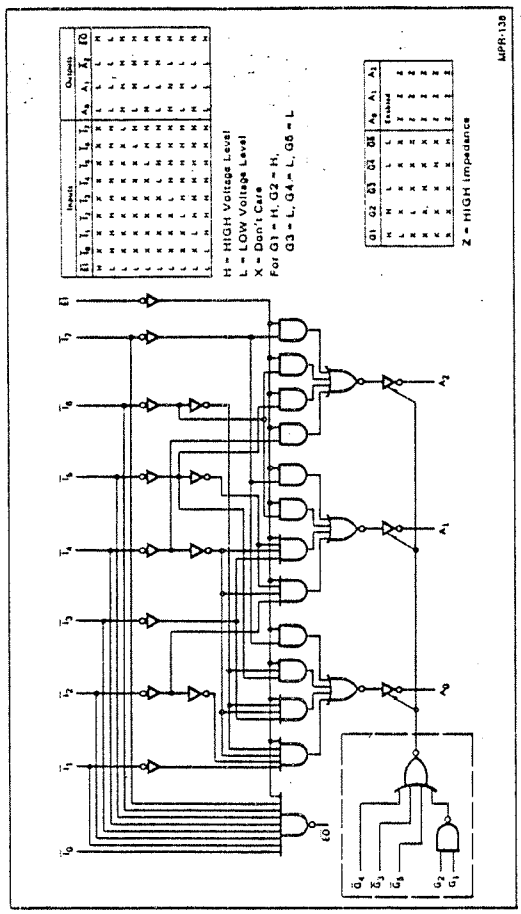


Figure 11. Am2913 Priority Interrupt Expander Logic Diagram and Truth Table.

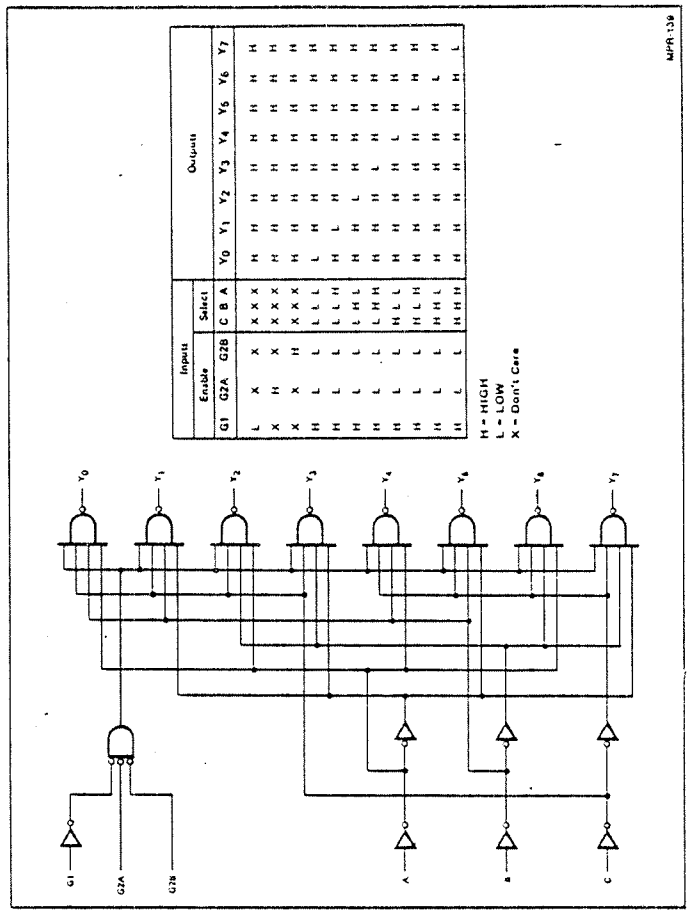


Figure Am25LS138 3 to 8 Line Decoder Logic Diagram and Truth Table.

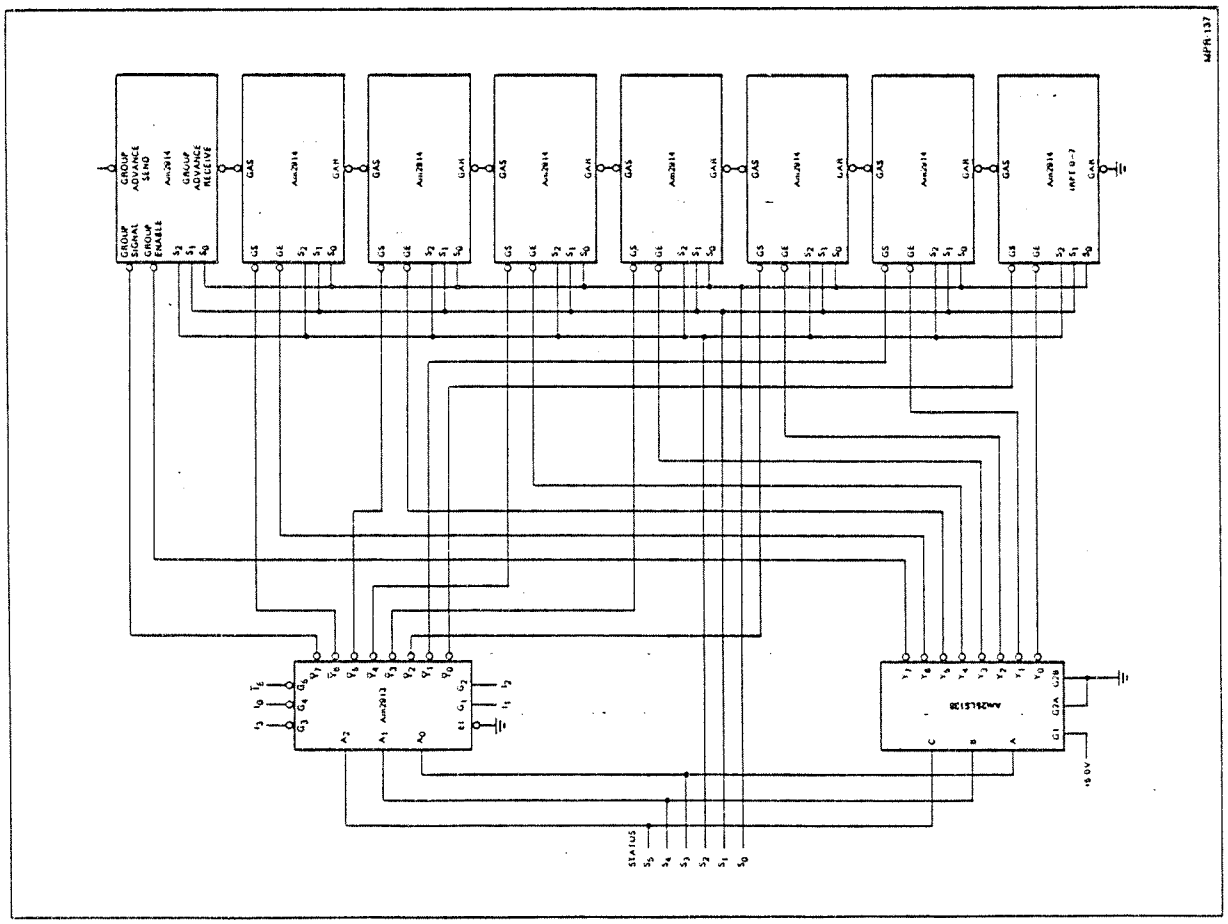


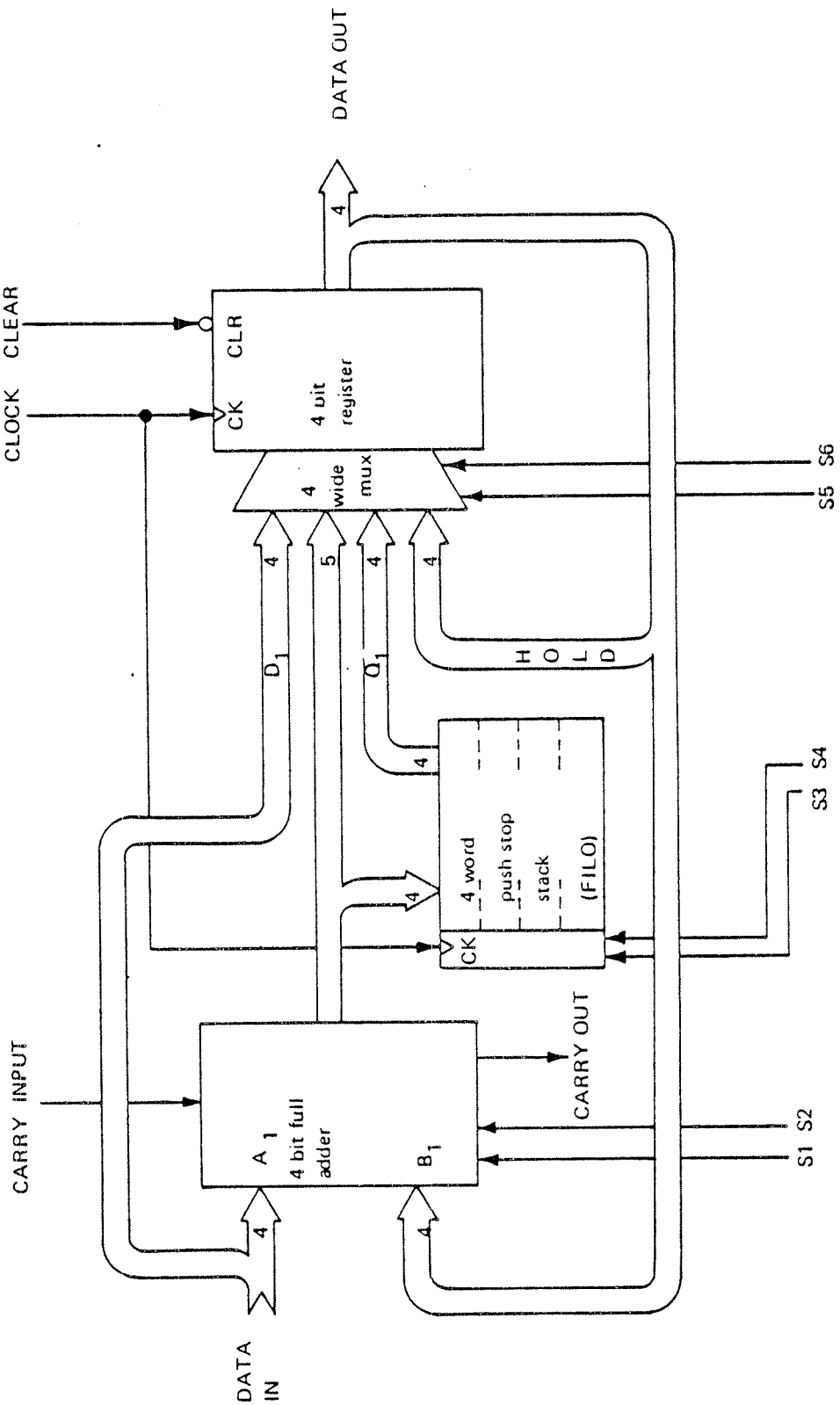
Figure 10. Group Signal, Group Enable, Group Advance Send, Group Advance Receive and Status Connections for Both the Parallel and Ripple Cascade M

APPENDIX D

FUNCTIONAL BLOCK DIAGRAM FOR 74S482 SEQUENCER

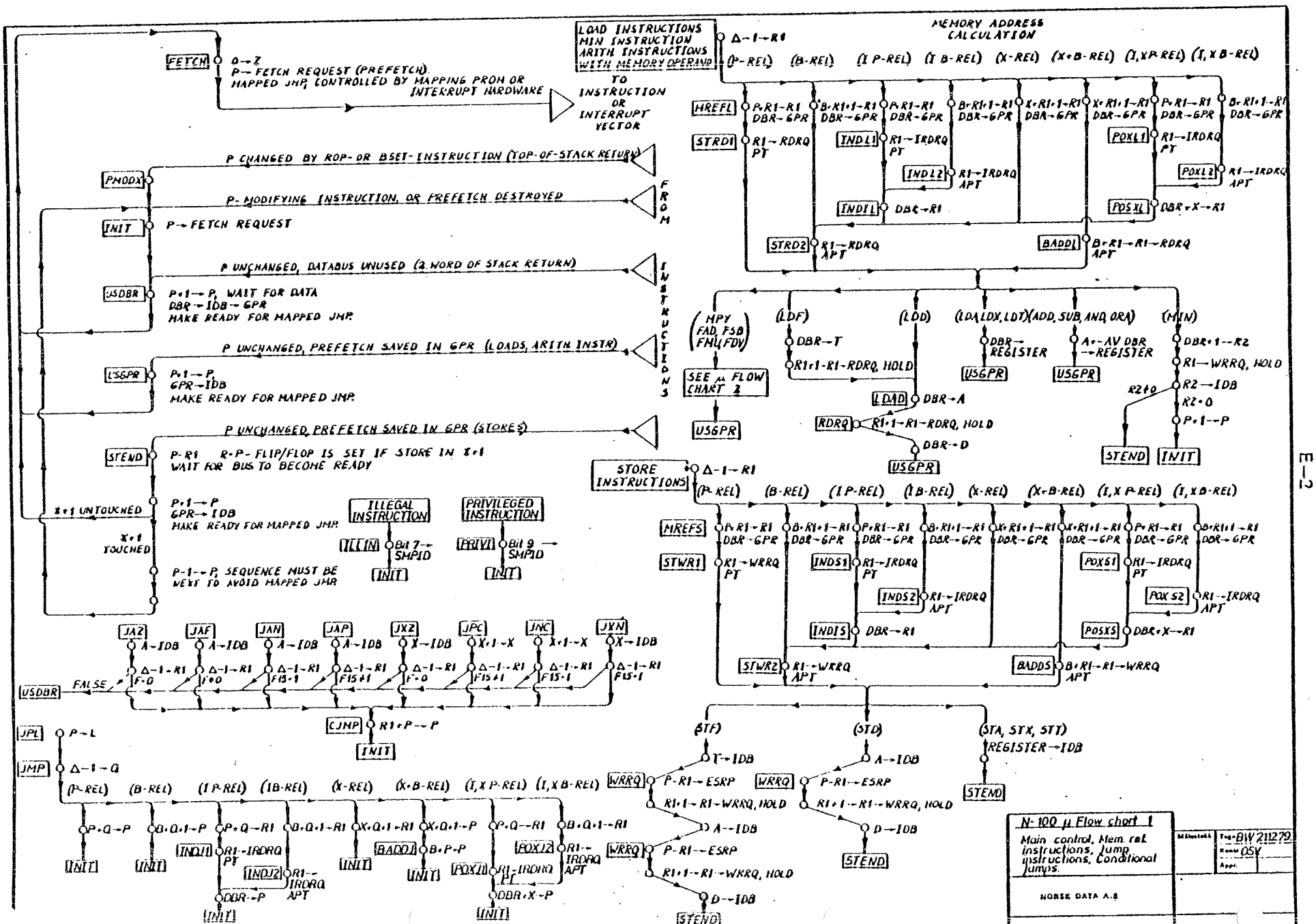
This appendix shows the functional block diagram for the 74S482 sequencer from Texas Instruments, Inc. More details will be found in "The TTL Data Book" issued by the same company.

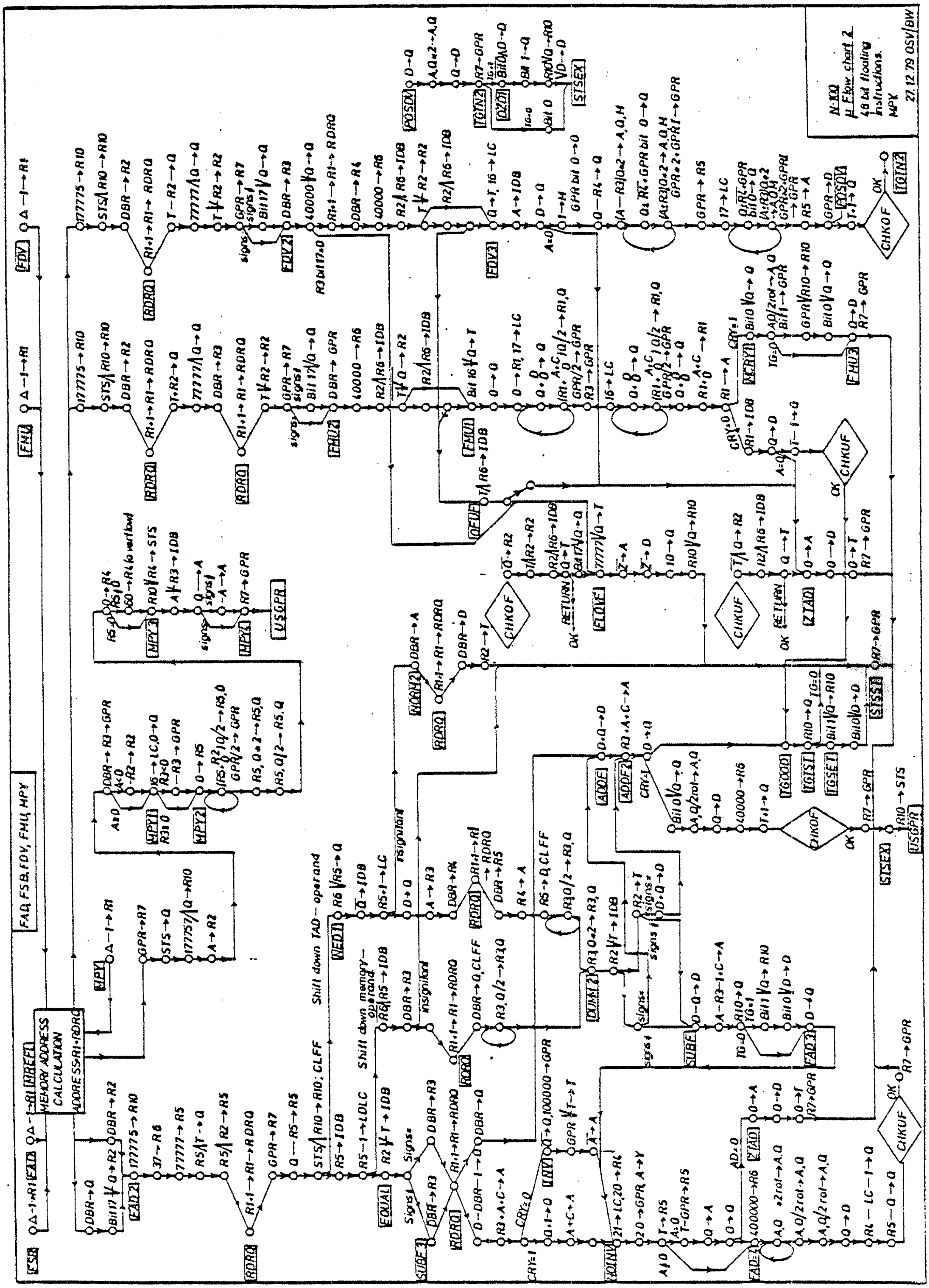
FUNCTIONAL BLOCK DIAGRAM

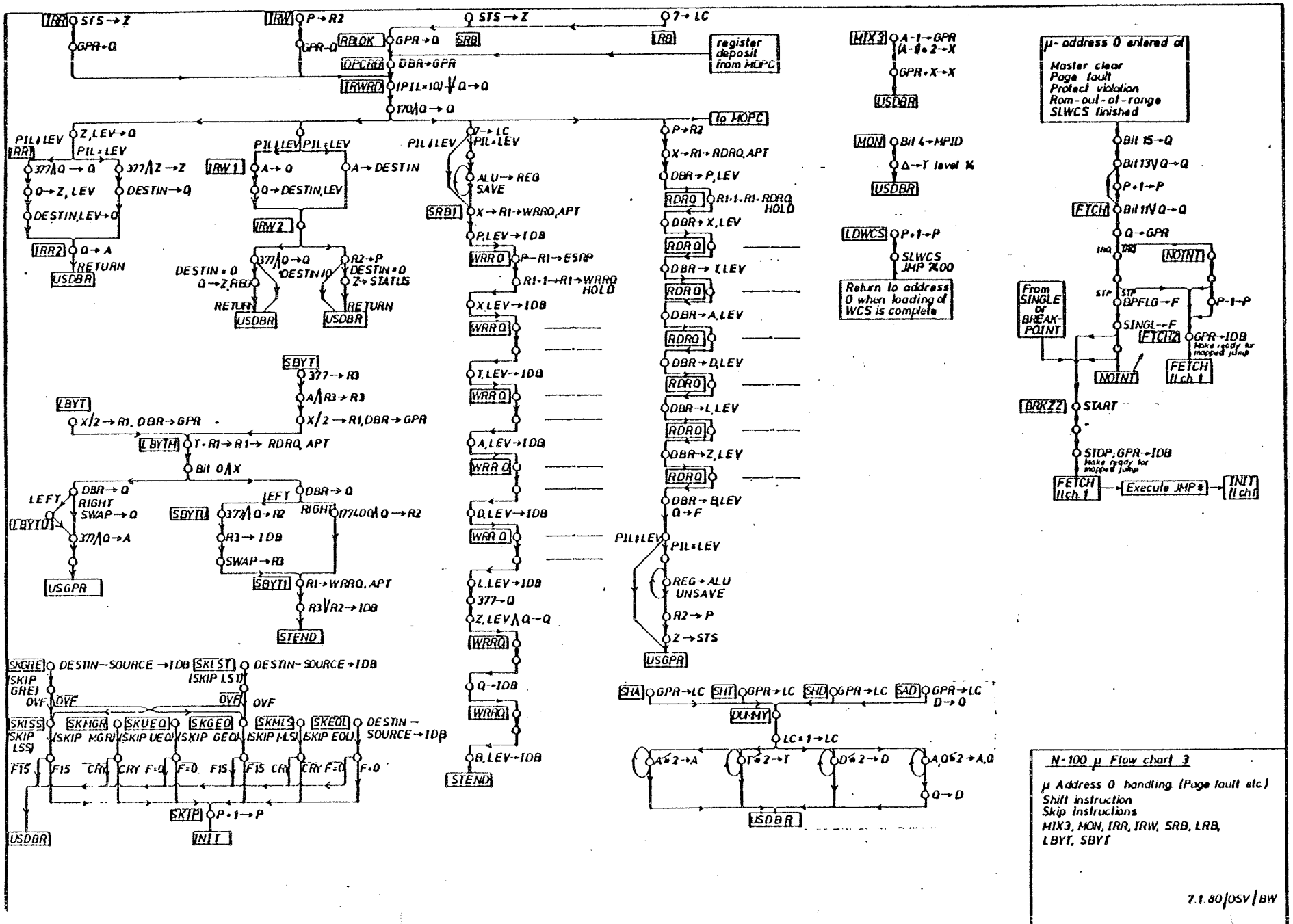


APPENDIX E

MICROPROGRAM FLOW CHARTS


FETCH
0→Z
P→ FETCH REQUEST (PREFETCH).
MAPPED JMP, CONTROLLED BY MAPPING PROM OR INTERRUPT HARDWARE
TO INSTRUCTION OR INTERRUPT VECTOR
LOAD INSTRUCTIONS
MIN INSTRUCTION
ARITH INSTRUCTIONS
WITH MEMORY OPERAND
Δ-1→R1
MEMORY ADDRESS CALCULATION
(P-REL) (B-REL) (I P-REL) (I B-REL) (X-REL) (X+B-REL) (I,X P-REL) (I,X B-REL)
P CHANGED BY ROP- OR BSET- INSTRUCTION (TOP-OF-STACK RETURN)
P- MODIFYING INSTRUCTION, OR PREFETCH DESTROYED
P→ FETCH REQUEST
P UNCHANGED, DATABUS UNUSED (2. WORD OF STACK RETURN)
P+1→ P, WAIT FOR DATA
DBR→ IDB→ GPR
MAKE READY FOR MAPPED JMP.
P UNCHANGED, PREFETCH SAVED IN GPR (LOADS, ARITH INSTR)
P+1→ P
GPR→IDB
MAKE READY FOR MAPPED JMP.
P UNCHANGED, PREFETCH SAVED IN GPR (STORES)
P-R1 R-P- FLIP/FLOP IS SET IF STORE IN X+1
WAIT FOR BUS TO BECOME READY
X+1 UNTOUCHED
P+1→ P
GPR→ IDB
MAKE READY FOR MAPPED JMP.
X+1 TOUCHED
P-1→ P, SEQUENCE MUST BE NEXT TO AVOID MAPPED JMP.
ILLEGAL INSTRUCTION
PRIVILEGED INSTRUCTION
FROM INSTRUCTIONS
MREFL
STRD1
STRD2
INDL1
INDL2
INDL1
POXL1
POXL2
POXL1
BADD1
MPY FAD, FSB FMU, FDV
SEE μ FLOW CHART 2
(LDF)
(LDD)
(LDA, LDX, LDT)
(ADD, SUB, AND, ORA)
(MIN)
LOAD
RDRQ
STEND
INIT
STORE INSTRUCTIONS
MREFS
STWR1
STWR2
INDS1
INDS2
INDS1
POXS1
POXS2
POXS1
BADDS
(STF)
(STD)
(STA, STX, STT)
WRRQ
STEND
JAZ
JAF
JAN
JAP
JXZ
JPC
JNC
JXN
USDBR
FALSE
CJMP
JPL
JMP
INDJ1
INDJ2
BADDJ
POXJ1
POXJ2
N-100 μ Flow chart 1
Main control, Mem. ref. instructions, Jump instructions, Conditional jumps.
NORSK DATA A.S
Title BW 211279
Name OSV
Appr.

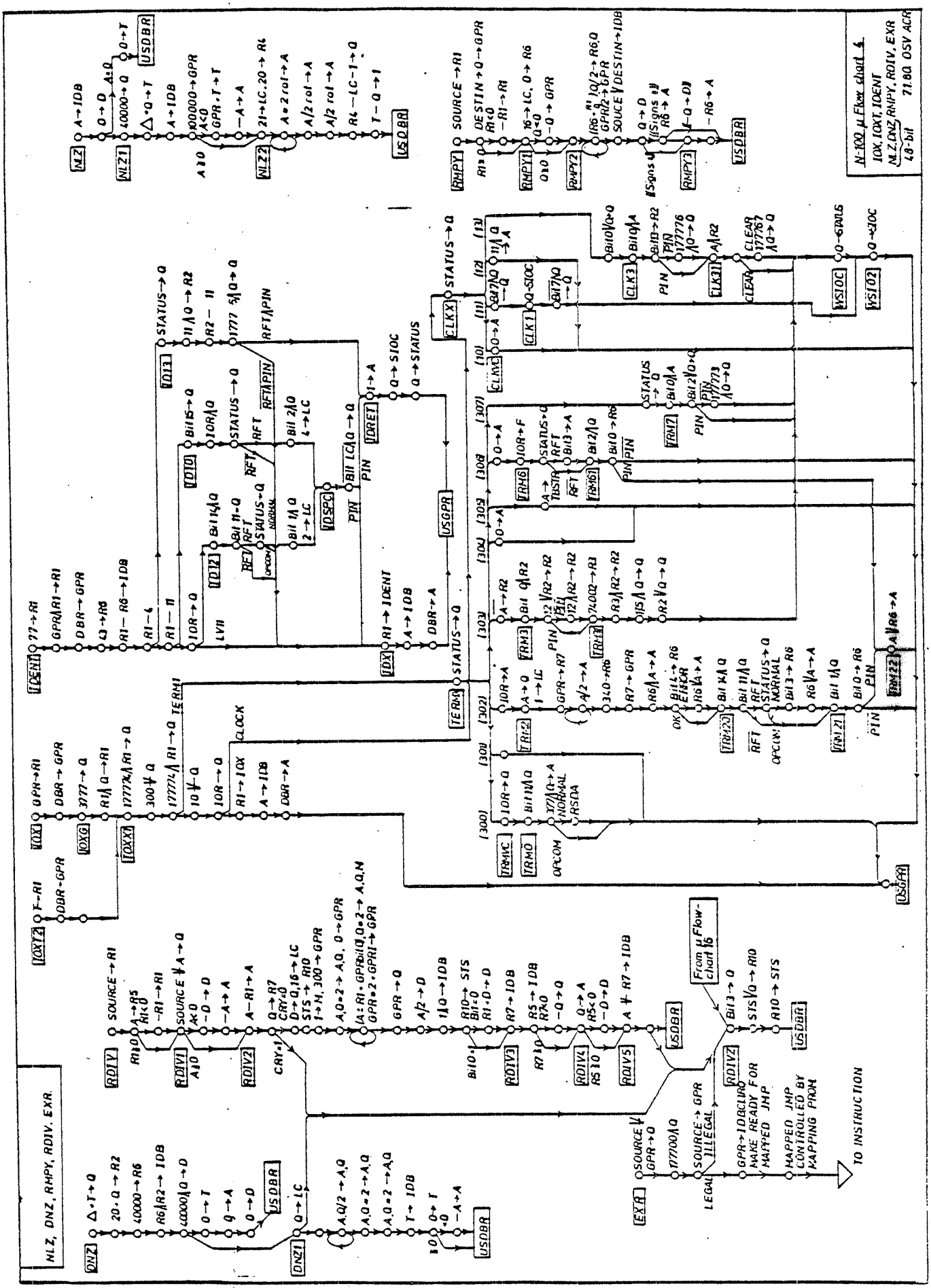

N-100
μ Flow chart 2
48 bit floating instructions.
MPY
27.12.79 OSV/BW
FAD, FSB, FDV, FMU, MPY
MEMORY ADDRESS CALCULATION
ADDRESS→R1→RDRQ

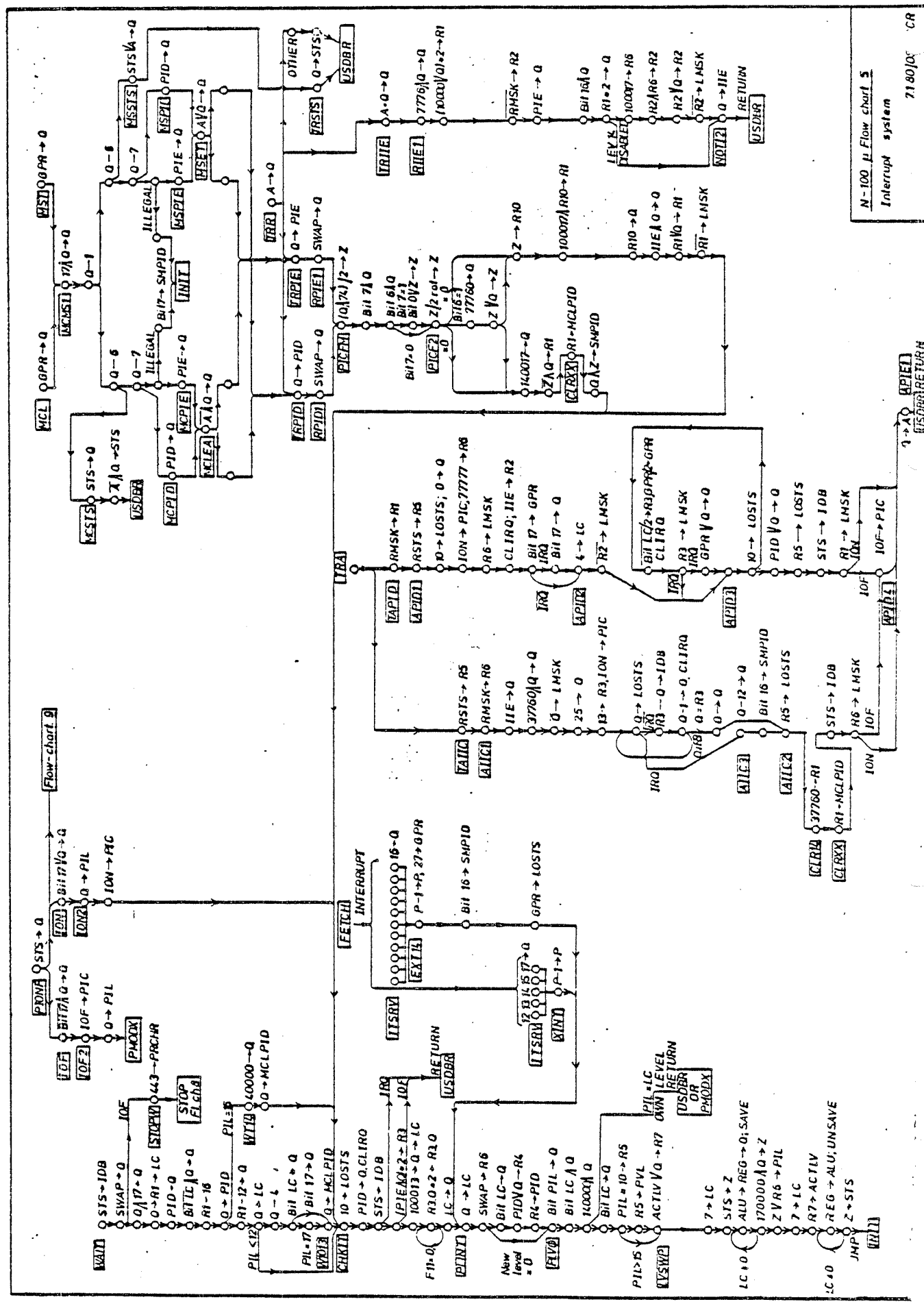


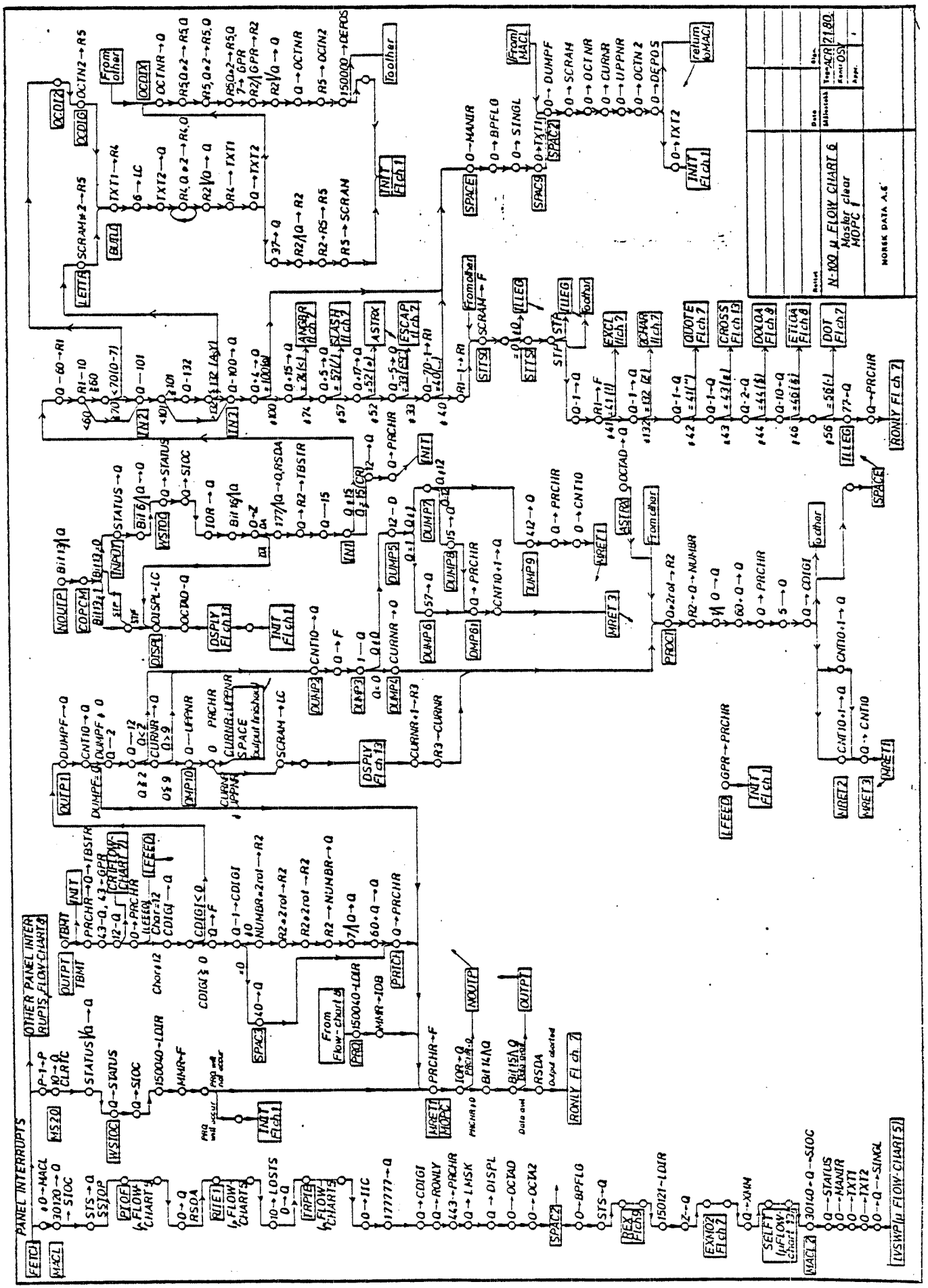
N-100 μ Flow chart 3

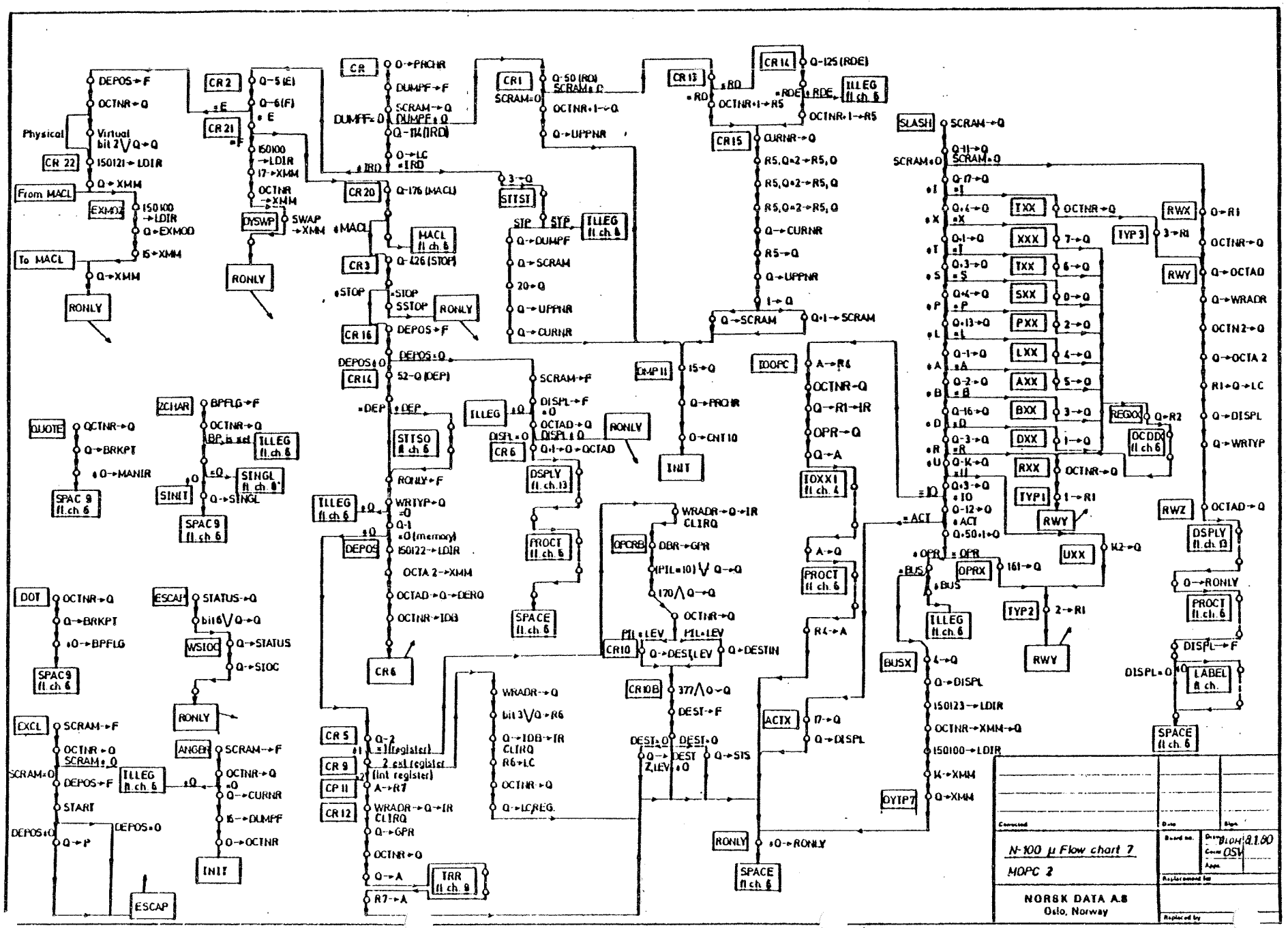
μ Address 0 handling (Page fault etc.)
Shift instruction
Skip instructions
MIX3, MON, IRR, IRW, SRB, LRB, LBYT, SBYT

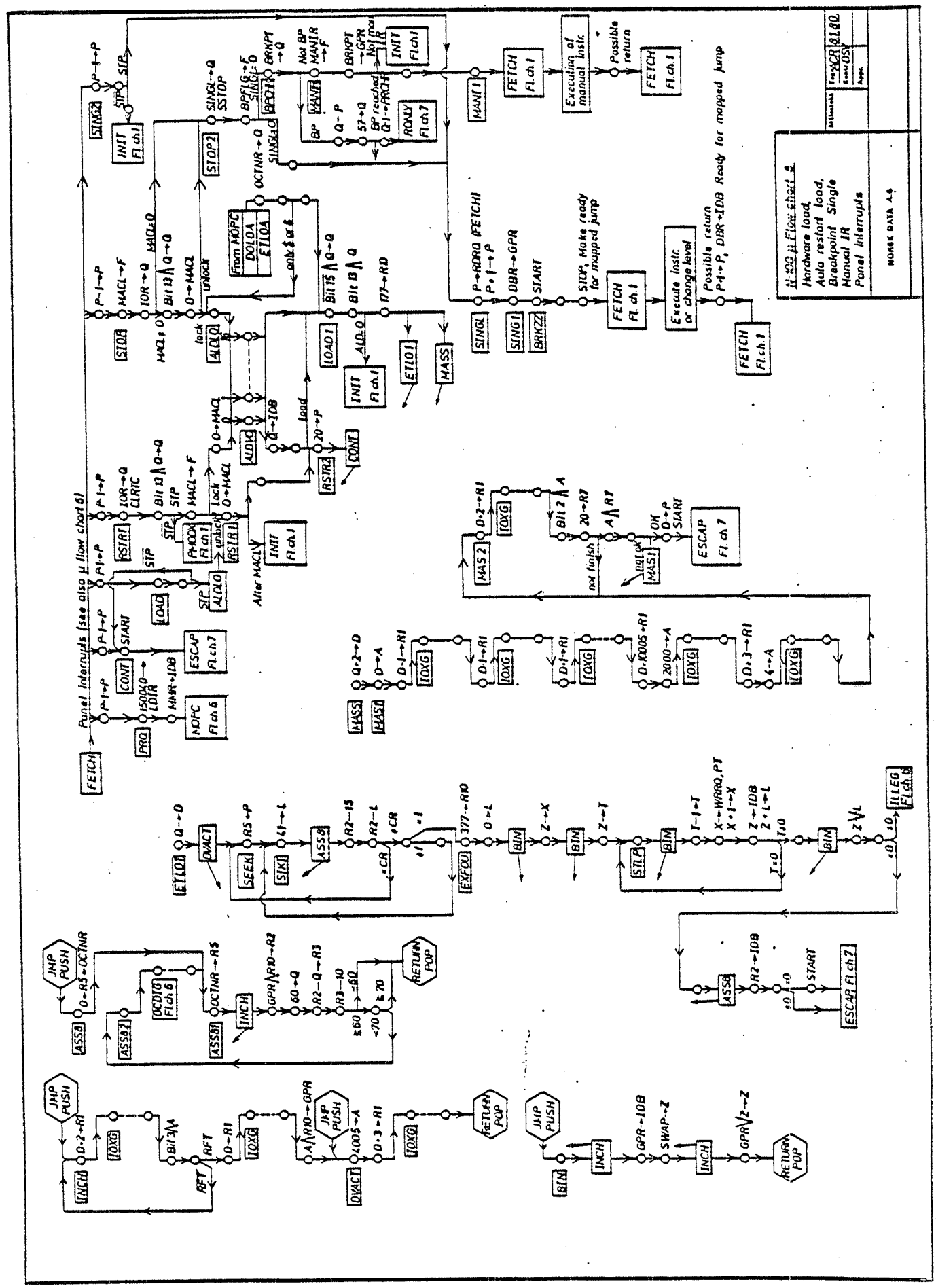
7.1.80/OSV/BW

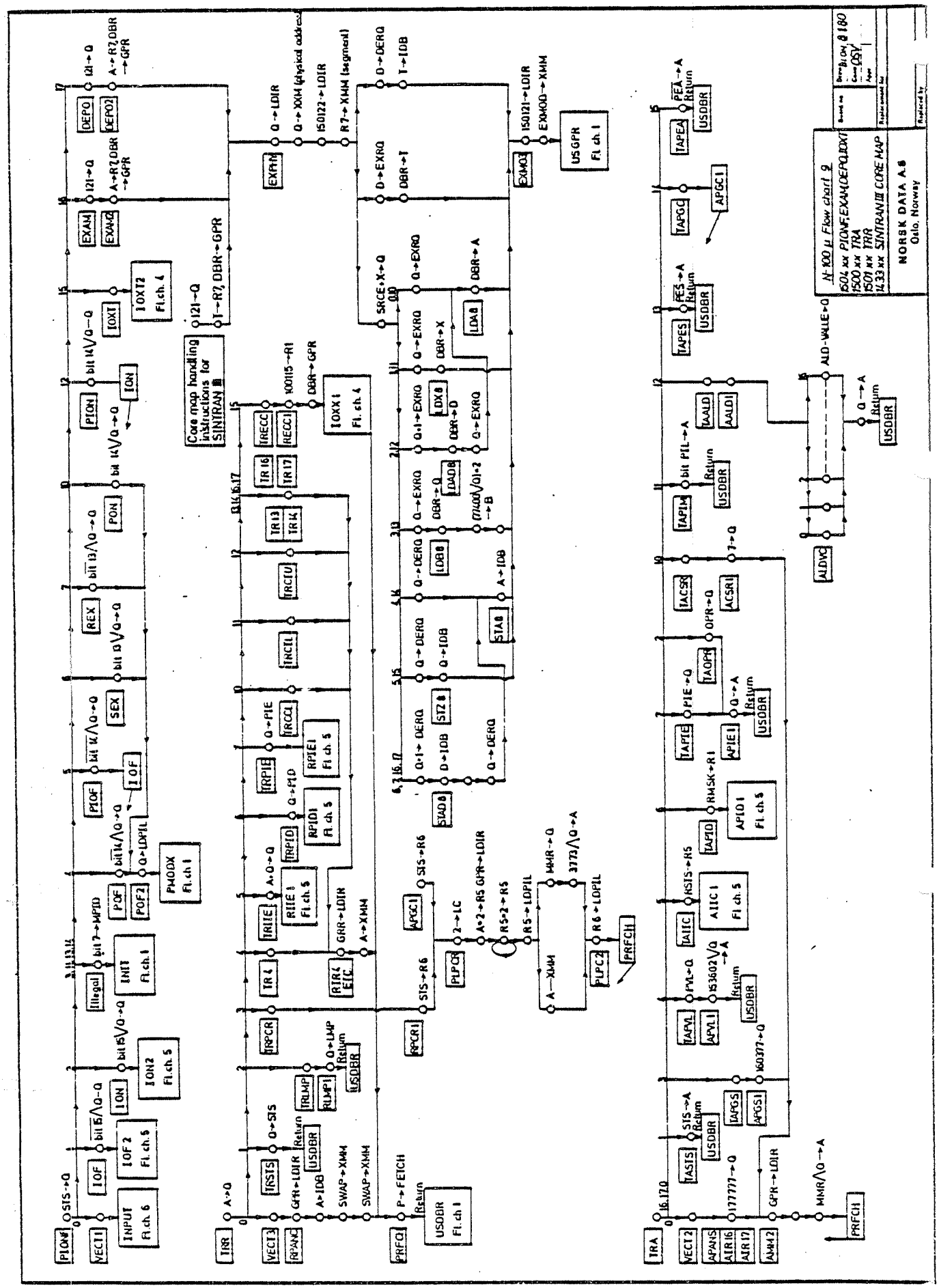

N-100 μ Flow chart 4.
10X, 10XT, IDENT
NLZ, DNZ, RMPY, RDIV, EXR
48-bit 7180 OSV ACR
NLZ, DNZ, RMPY, RDIV, EXR.
TO INSTRUCTION
From μ Flow-chart 16
MAKE READY FOR MAPPED JMP
MAPPED JMP CONTROLLED BY MAPPING PROM
ILLEGAL
OPCOM
NORMAL
RFT
PIN
CLEAR
CLOCK

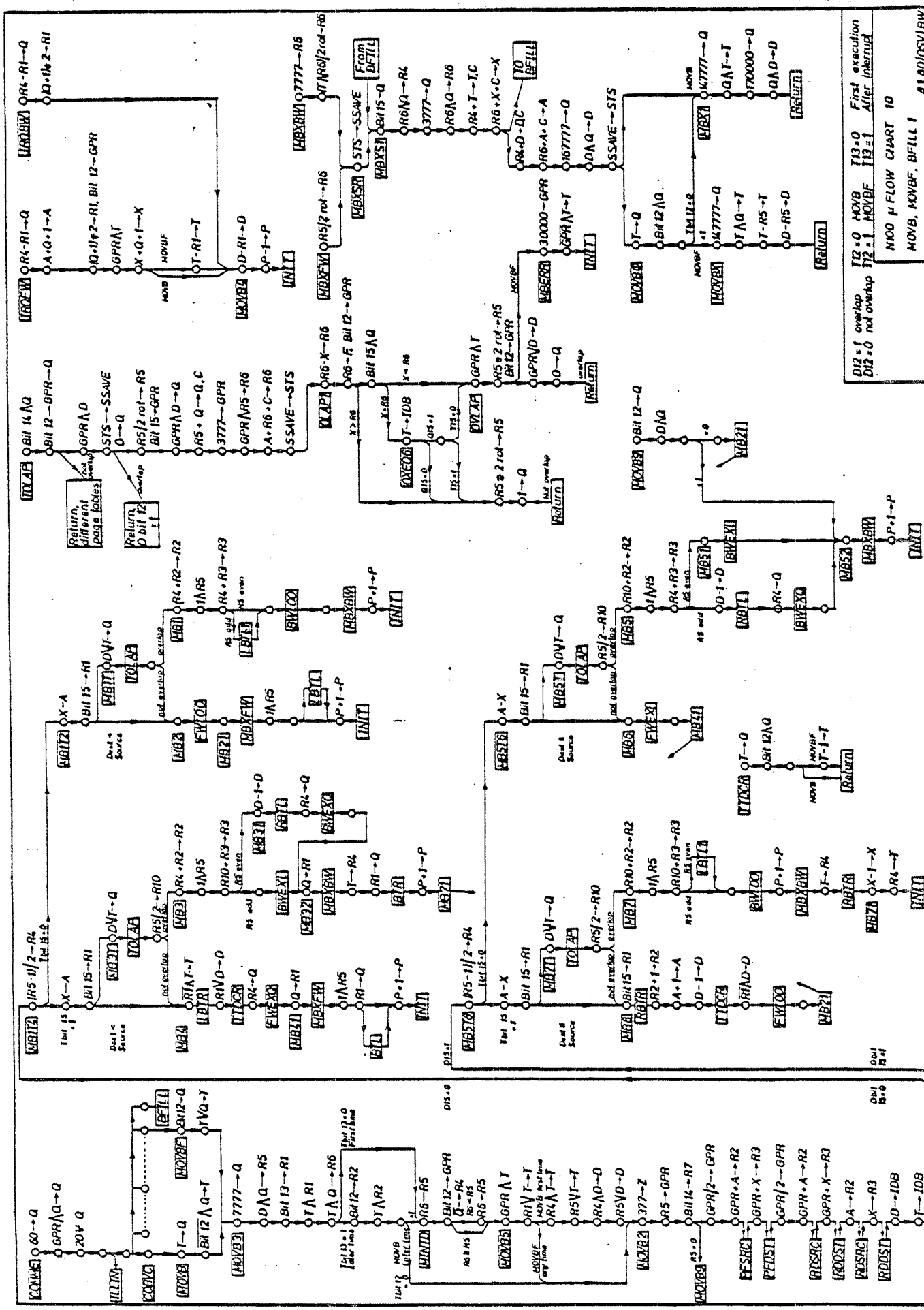

N-100 µ Flow chart 5
Interrupt system
7180/0
Flow-chart 9
INTERRUPT
FETCH
RETURN
WAIT
CHK11
WDT3
WDT2
WDT1
PMODX
STOP
FLG68
STOPID
INIT
ILLEGAL
UNIT
VSPID
VSSTS
VSET
MCPIE
MCLEA
MCPID
MCSTS
MCMS1
MCL
VAPID
VAIC1
VAIC2
VAIC3
VAIC4
VAIC5
VAIC6
VAIC7
VAIC8
VAIC9
VAIC10
VAIC11
VAIC12
VAIC13
VAIC14
VAIC15
VAIC16
VAIC17
VAIC18
VAIC19
VAIC20
VAIC21
VAIC22
VAIC23
VAIC24
VAIC25
VAIC26
VAIC27
VAIC28
VAIC29
VAIC30
VAIC31
VAIC32
VAIC33
VAIC34
VAIC35
VAIC36
VAIC37
VAIC38
VAIC39
VAIC40
VAIC41
VAIC42
VAIC43
VAIC44
VAIC45
VAIC46
VAIC47
VAIC48
VAIC49
VAIC50
VAIC51
VAIC52
VAIC53
VAIC54
VAIC55
VAIC56
VAIC57
VAIC58
VAIC59
VAIC60
VAIC61
VAIC62
VAIC63
VAIC64
VAIC65
VAIC66
VAIC67
VAIC68
VAIC69
VAIC70
VAIC71
VAIC72
VAIC73
VAIC74
VAIC75
VAIC76
VAIC77
VAIC78
VAIC79
VAIC80
VAIC81
VAIC82
VAIC83
VAIC84
VAIC85
VAIC86
VAIC87
VAIC88
VAIC89
VAIC90
VAIC91
VAIC92
VAIC93
VAIC94
VAIC95
VAIC96
VAIC97
VAIC98
VAIC99
VAIC100
VAIC101
VAIC102
VAIC103
VAIC104
VAIC105
VAIC106
VAIC107
VAIC108
VAIC109
VAIC110
VAIC111
VAIC112
VAIC113
VAIC114
VAIC115
VAIC116
VAIC117
VAIC118
VAIC119
VAIC120
VAIC121
VAIC122
VAIC123
VAIC124
VAIC125
VAIC126
VAIC127
VAIC128
VAIC129
VAIC130
VAIC131
VAIC132
VAIC133
VAIC134
VAIC135
VAIC136
VAIC137
VAIC138
VAIC139
VAIC140
VAIC141
VAIC142
VAIC143
VAIC144
VAIC145
VAIC146
VAIC147
VAIC148
VAIC149
VAIC150
VAIC151
VAIC152
VAIC153
VAIC154
VAIC155
VAIC156
VAIC157
VAIC158
VAIC159
VAIC160
VAIC161
VAIC162
VAIC163
VAIC164
VAIC165
VAIC166
VAIC167
VAIC168
VAIC169
VAIC170
VAIC171
VAIC172
VAIC173
VAIC174
VAIC175
VAIC176
VAIC177
VAIC178
VAIC179
VAIC180
VAIC181
VAIC182
VAIC183
VAIC184
VAIC185
VAIC186
VAIC187
VAIC188
VAIC189
VAIC190
VAIC191
VAIC192
VAIC193
VAIC194
VAIC195
VAIC196
VAIC197
VAIC198
VAIC199
VAIC200
VAIC201
VAIC202
VAIC203
VAIC204
VAIC205
VAIC206
VAIC207
VAIC208
VAIC209
VAIC210
VAIC211
VAIC212
VAIC213
VAIC214
VAIC215
VAIC216
VAIC217
VAIC218
VAIC219
VAIC220
VAIC221
VAIC222
VAIC223
VAIC224
VAIC225
VAIC226
VAIC227
VAIC228
VAIC229
VAIC230
VAIC231
VAIC232
VAIC233
VAIC234
VAIC235
VAIC236
VAIC237
VAIC238
VAIC239
VAIC240
VAIC241
VAIC242
VAIC243
VAIC244
VAIC245
VAIC246
VAIC247
VAIC248
VAIC249
VAIC250
VAIC251
VAIC252
VAIC253
VAIC254
VAIC255
VAIC256
VAIC257
VAIC258
VAIC259
VAIC260
VAIC261
VAIC262
VAIC263
VAIC264
VAIC265
VAIC266
VAIC267
VAIC268
VAIC269
VAIC270
VAIC271
VAIC272
VAIC273
VAIC274
VAIC275
VAIC276
VAIC277
VAIC278
VAIC279
VAIC280
VAIC281
VAIC282
VAIC283
VAIC284
VAIC285
VAIC286
VAIC287
VAIC288
VAIC289
VAIC290
VAIC291
VAIC292
VAIC293
VAIC294
VAIC295
VAIC296
VAIC297
VAIC298
VAIC299
VAIC300
VAIC301
VAIC302
VAIC303
VAIC304
VAIC305
VAIC306
VAIC307
VAIC308
VAIC309
VAIC310
VAIC311
VAIC312
VAIC313
VAIC314
VAIC315
VAIC316
VAIC317
VAIC318
VAIC319
VAIC320
VAIC321
VAIC322
VAIC323
VAIC324
VAIC325
VAIC326
VAIC327
VAIC328
VAIC329
VAIC330
VAIC331
VAIC332
VAIC333
VAIC334
VAIC335
VAIC336
VAIC337
VAIC338
VAIC339
VAIC340
VAIC341
VAIC342
VAIC343
VAIC344
VAIC345
VAIC346
VAIC347
VAIC348
VAIC349
VAIC350
VAIC351
VAIC352
VAIC353
VAIC354
VAIC355
VAIC356
VAIC357
VAIC358
VAIC359
VAIC360
VAIC361
VAIC362
VAIC363
VAIC364
VAIC365
VAIC366
VAIC367
VAIC368
VAIC369
VAIC370
VAIC371
VAIC372
VAIC373
VAIC374
VAIC375
VAIC376
VAIC377
VAIC378
VAIC379
VAIC380
VAIC381
VAIC382
VAIC383
VAIC384
VAIC385
VAIC386
VAIC387
VAIC388
VAIC389
VAIC390
VAIC391
VAIC392
VAIC393
VAIC394
VAIC395
VAIC396
VAIC397
VAIC398
VAIC399
VAIC400
VAIC401
VAIC402
VAIC403
VAIC404
VAIC405
VAIC406
VAIC407
VAIC408
VAIC409
VAIC410
VAIC411
VAIC412
VAIC413
VAIC414
VAIC415
VAIC416
VAIC417
VAIC418
VAIC419
VAIC420
VAIC421
VAIC422
VAIC423
VAIC424
VAIC425
VAIC426
VAIC427
VAIC428
VAIC429
VAIC430
VAIC431
VAIC432
VAIC433
VAIC434
VAIC435
VAIC436
VAIC437
VAIC438
VAIC439
VAIC440
VAIC441
VAIC442
VAIC443
VAIC444
VAIC445
VAIC446
VAIC447
VAIC448
VAIC449
VAIC450
VAIC451
VAIC452
VAIC453
VAIC454
VAIC455
VAIC456
VAIC457
VAIC458
VAIC459
VAIC460
VAIC461
VAIC462
VAIC463
VAIC464
VAIC465
VAIC466
VAIC467
VAIC468
VAIC469
VAIC470
VAIC471
VAIC472
VAIC473
VAIC474
VAIC475
VAIC476
VAIC477
VAIC478
VAIC479
VAIC480
VAIC481
VAIC482
VAIC483
VAIC484
VAIC485
VAIC486
VAIC487
VAIC488
VAIC489
VAIC490
VAIC491
VAIC492
VAIC493
VAIC494
VAIC495
VAIC496
VAIC497
VAIC498
VAIC499
VAIC500
VAIC501
VAIC502
VAIC503
VAIC504
VAIC505
VAIC506
VAIC507
VAIC508
VAIC509
VAIC510
VAIC511
VAIC512
VAIC513
VAIC514
VAIC515
VAIC516
VAIC517
VAIC518
VAIC519
VAIC520
VAIC521
VAIC522
VAIC523
VAIC524
VAIC525
VAIC526
VAIC527
VAIC528
VAIC529
VAIC530
VAIC531
VAIC532
VAIC533
VAIC534
VAIC535
VAIC536
VAIC537
VAIC538
VAIC539
VAIC540
VAIC541
VAIC542
VAIC543
VAIC544
VAIC545
VAIC546
VAIC547
VAIC548
VAIC549
VAIC550
VAIC551
VAIC552
VAIC553
VAIC554
VAIC555
VAIC556
VAIC557
VAIC558
VAIC559
VAIC560
VAIC561
VAIC562
VAIC563
VAIC564
VAIC565
VAIC566
VAIC567
VAIC568
VAIC569
VAIC570
VAIC571
VAIC572
VAIC573
VAIC574
VAIC575
VAIC576
VAIC577
VAIC578
VAIC579
VAIC580
VAIC581
VAIC582
VAIC583
VAIC584
VAIC585
VAIC586
VAIC587
VAIC588
VAIC589
VAIC590
VAIC591
VAIC592
VAIC593
VAIC594
VAIC595
VAIC596
VAIC597
VAIC598
VAIC599
VAIC600
VAIC601
VAIC602
VAIC603
VAIC604
VAIC605
VAIC606
VAIC607
VAIC608
VAIC609
VAIC610
VAIC611
VAIC612
VAIC613
VAIC614
VAIC615
VAIC616
VAIC617
VAIC618
VAIC619
VAIC620
VAIC621
VAIC622
VAIC623
VAIC624
VAIC625
VAIC626
VAIC627
VAIC628
VAIC629
VAIC630
VAIC631
VAIC632
VAIC633
VAIC634
VAIC635
VAIC636
VAIC637
VAIC638
VAIC639
VAIC640
VAIC641
VAIC642
VAIC643
VAIC644
VAIC645
VAIC646
VAIC647
VAIC648
VAIC649
VAIC650
VAIC651
VAIC652
VAIC653
VAIC654
VAIC655
VAIC656
VAIC657
VAIC658
VAIC659
VAIC660
VAIC661
VAIC662
VAIC663
VAIC664
VAIC665
VAIC666
VAIC667
VAIC668
VAIC669
VAIC670
VAIC671
VAIC672
VAIC673
VAIC674
VAIC675
VAIC676
VAIC677
VAIC678
VAIC679
VAIC680
VAIC681
VAIC682
VAIC683
VAIC684
VAIC685
VAIC686
VAIC687
VAIC688
VAIC689
VAIC690
VAIC691
VAIC692
VAIC693
VAIC694
VAIC695
VAIC696
VAIC697
VAIC698
VAIC699
VAIC700
VAIC701
VAIC702
VAIC703
VAIC704
VAIC705
VAIC706
VAIC707
VAIC708
VAIC709
VAIC710
VAIC711
VAIC712
VAIC713
VAIC714
VAIC715
VAIC716
VAIC717
VAIC718
VAIC719
VAIC720
VAIC721
VAIC722
VAIC723
VAIC724
VAIC725
VAIC726
VAIC727
VAIC728
VAIC729
VAIC730
VAIC731
VAIC732
VAIC733
VAIC734
VAIC735
VAIC736
VAIC737
VAIC738
VAIC739
VAIC740
VAIC741
VAIC742
VAIC743
VAIC744
VAIC745
VAIC746
VAIC747
VAIC748
VAIC749
VAIC750
VAIC751
VAIC752
VAIC753
VAIC754
VAIC755
VAIC756
VAIC757
VAIC758
VAIC759
VAIC760
VAIC761
VAIC762
VAIC763
VAIC764
VAIC765
VAIC766
VAIC767
VAIC768
VAIC769
VAIC770
VAIC771
VAIC772
VAIC773
VAIC774
VAIC775
VAIC776
VAIC777
VAIC778
VAIC779
VAIC780
VAIC781
VAIC782
VAIC783
VAIC784
VAIC785
VAIC786
VAIC787
VAIC788
VAIC789
VAIC790
VAIC791
VAIC792
VAIC793
VAIC794
VAIC795
VAIC796
VAIC797
VAIC798
VAIC799
VAIC800
VAIC801
VAIC802
VAIC803
VAIC804
VAIC805
VAIC806
VAIC807
VAIC808
VAIC809
VAIC810
VAIC811
VAIC812
VAIC813
VAIC814
VAIC815
VAIC816
VAIC817
VAIC818
VAIC819
VAIC820
VAIC821
VAIC822
VAIC823
VAIC824
VAIC825
VAIC826
VAIC827
VAIC828
VAIC829
VAIC830
VAIC831
VAIC832
VAIC833
VAIC834
VAIC835
VAIC836
VAIC837
VAIC838
VAIC839
VAIC840
VAIC841
VAIC842
VAIC843
VAIC844
VAIC845
VAIC846
VAIC847
VAIC848
VAIC849
VAIC850
VAIC851
VAIC852
VAIC853
VAIC854
VAIC855
VAIC856
VAIC857
VAIC858
VAIC859
VAIC860
VAIC861
VAIC862
VAIC863
VAIC864
VAIC865
VAIC866
VAIC867
VAIC868
VAIC869
VAIC870
VAIC871
VAIC872
VAIC873
VAIC874
VAIC875
VAIC876
VAIC877
VAIC878
VAIC879
VAIC880
VAIC881
VAIC882
VAIC883
VAIC884
VAIC885
VAIC886
VAIC887
VAIC888
VAIC889
VAIC890
VAIC891
VAIC892
VAIC893
VAIC894
VAIC895
VAIC896
VAIC897
VAIC898
VAIC899
VAIC900
VAIC901
VAIC902
VAIC903
VAIC904
VAIC905
VAIC906
VAIC907
VAIC908
VAIC909
VAIC910
VAIC911
VAIC912
VAIC913
VAIC914
VAIC915
VAIC916
VAIC917
VAIC918
VAIC919
VAIC920
VAIC921
VAIC922
VAIC923
VAIC924
VAIC925
VAIC926
VAIC927
VAIC928
VAIC929
VAIC930
VAIC931
VAIC932
VAIC933
VAIC934
VAIC935
VAIC936
VAIC937
VAIC938
VAIC939
VAIC940
VAIC941
VAIC942
VAIC943
VAIC944
VAIC945
VAIC946
VAIC947
VAIC948
VAIC949
VAIC950
VAIC951
VAIC952
VAIC953
VAIC954
VAIC955
VAIC956
VAIC957
VAIC958
VAIC959
VAIC960
VAIC961
VAIC962
VAIC963
VAIC964
VAIC965
VAIC966
VAIC967
VAIC968
VAIC969
VAIC970
VAIC971
VAIC972
VAIC973
VAIC974
VAIC975
VAIC976
VAIC977
VAIC978
VAIC979
VAIC980
VAIC981
VAIC982
VAIC983
VAIC984
VAIC985
VAIC986
VAIC987
VAIC988
VAIC989
VAIC990
VAIC991
VAIC992
VAIC993
VAIC994
VAIC995
VAIC996
VAIC997
VAIC998
VAIC999
VAIC1000

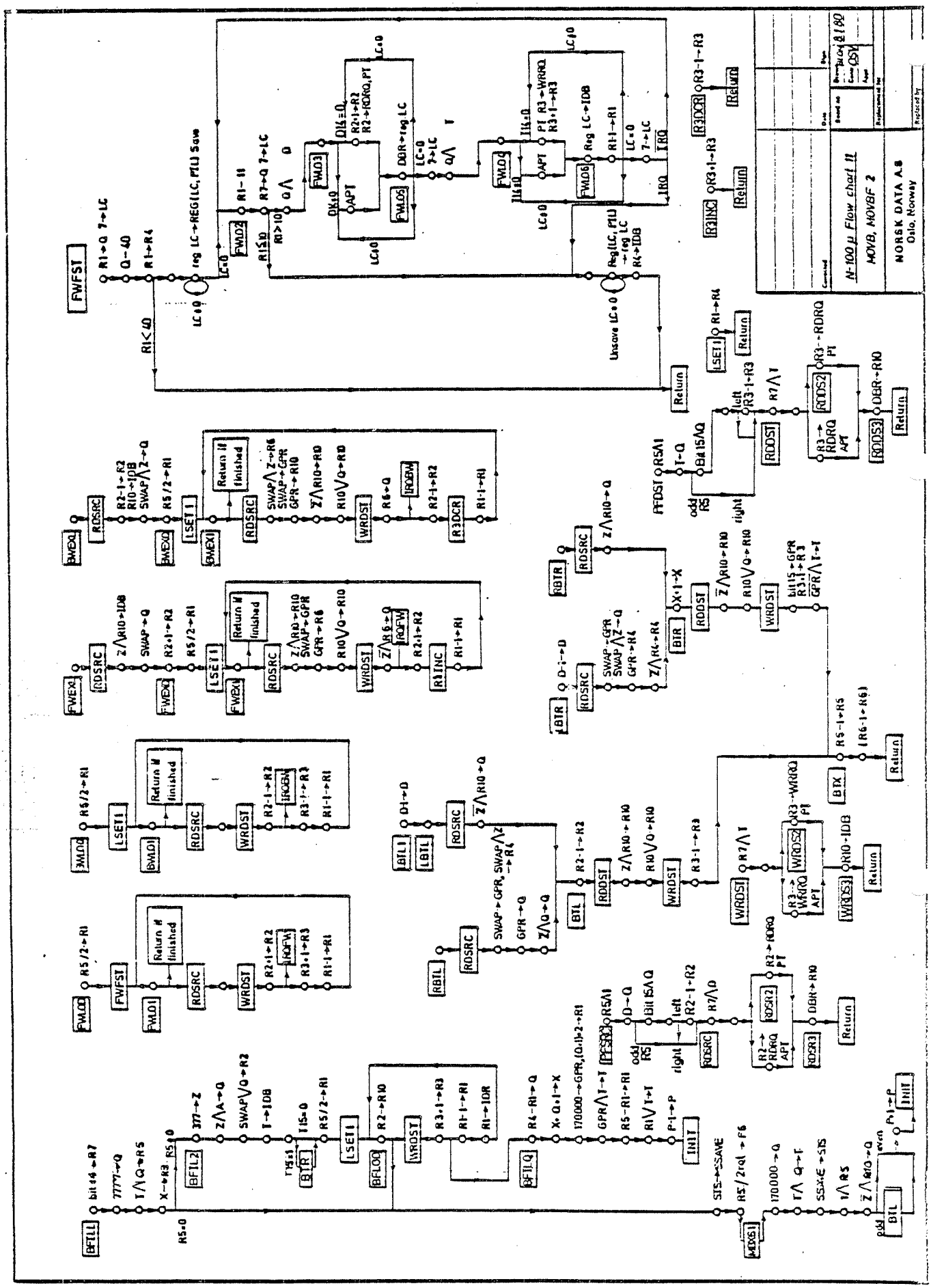

PANEL INTERRUPTS
OTHER PANEL INTER
RUPTS FLOW-CHART 6
N-100 μ FLOW CHART 6
Master clear
MOPC 1
NORSK DATA A.S
FEICI
MACL
MS20
WSTOC
PTOE
RITE1
TAPIE
REXT
EXNO2
SELFT
LYSWP
INIT
OUTPT
TBMT
RONLY
MRET1
MRET2
MRET3
ILEEG
ILEEG2
PRCH
DUMP1
DUMP2
DUMP3
DUMP4
DUMP5
DUMP6
DUMP8
DUMP9
DUMP10
DUMP11
DUMP12
DUMP13
DISPL
DSPLY
NOUTP
INPUT
VSTOC
VXIT
VIN1
VIN2
LETTR
BUILT
SPACE
SPAC2
SPAC3
SPAC4
SPAC5
SPACE
STTS1
STTS2
STTS3
STTS4
EXCL
QUOTE
CROSS
DOLLA
ETCAA
DOT
ASTRX
ESCAP
ANGBR
SLASH
QCHAR
OCDIX
OCDI2
OCDI6
OCTN2
From other
From MACL
return MACL
Toother
Fromother
Fromchar
Tochar
Top-ACR 21.80
Date
Initials
Sign
Appr.
Rev.
Page
Date
Initials
Sign
Appr.
Rev.
Page
ND-06.018.01

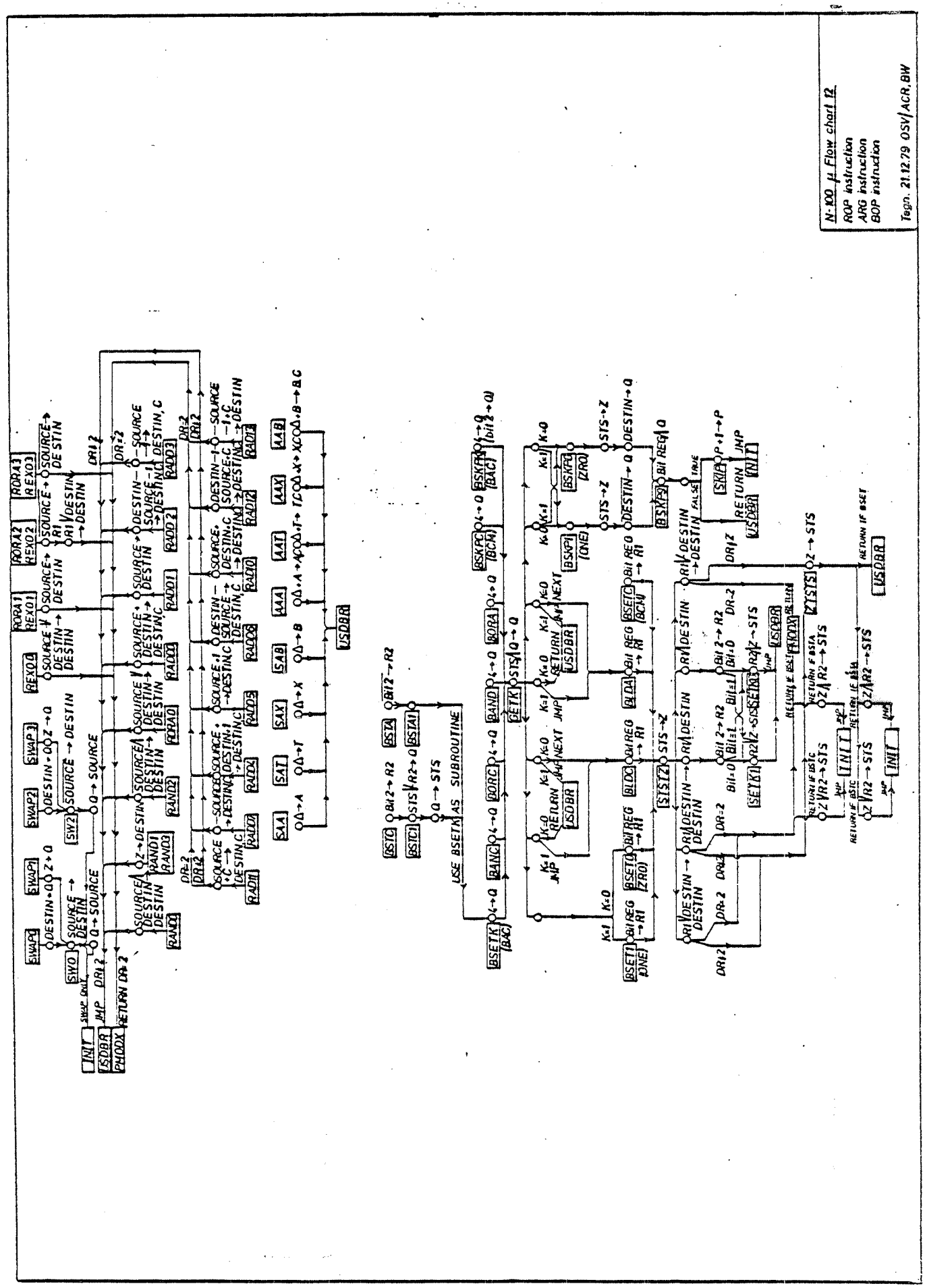

N-100 μ Flow chart 7
MOPC 2
NORSK DATA A.S
Oslo, Norway
ND-06.018.01

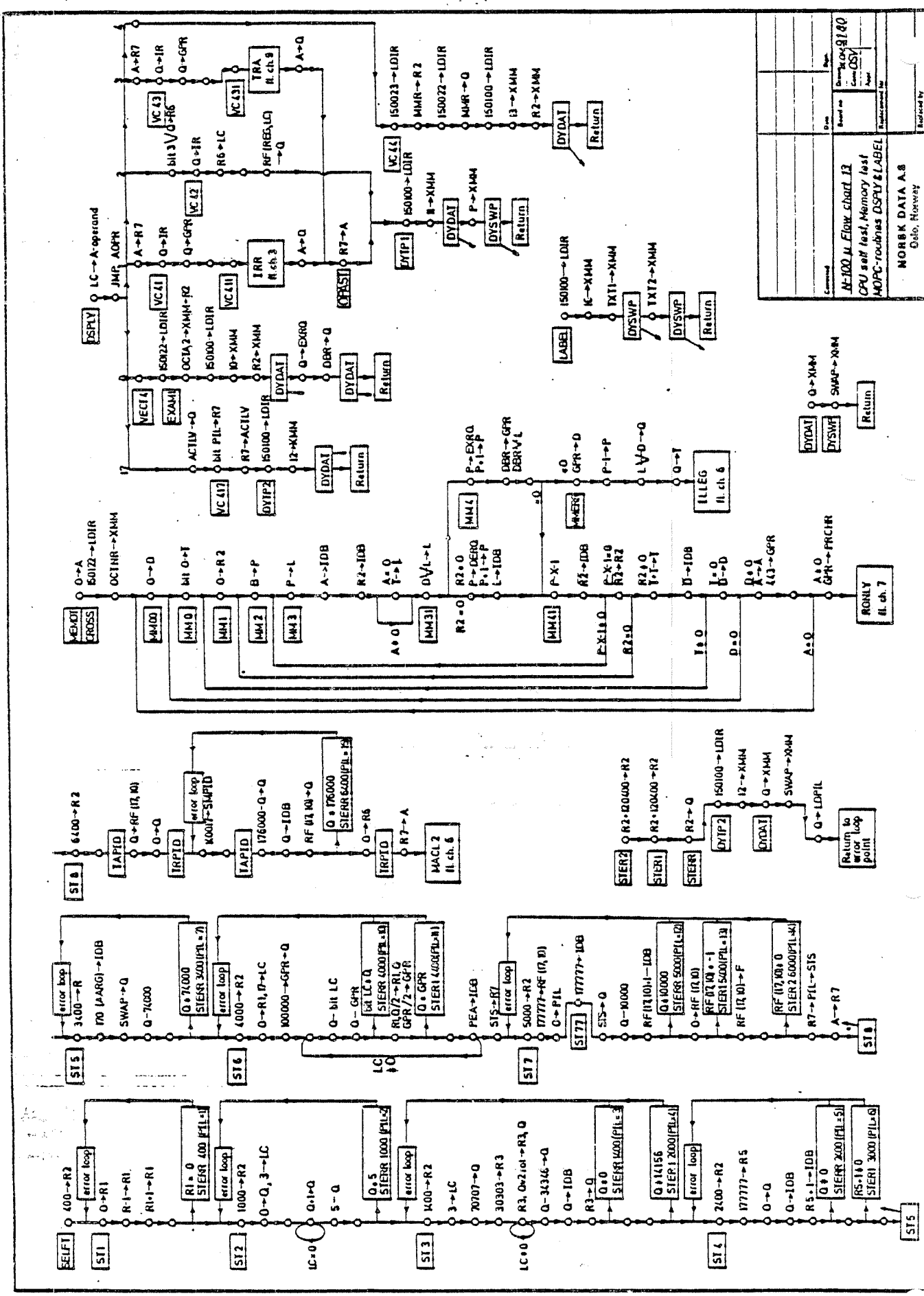

Panel interrupts (see also µ flow chart 6)
Hardware load,
Auto restart load,
Breakpoint Single
Manual IR
Panel interrupts
NORSK DATA A.S

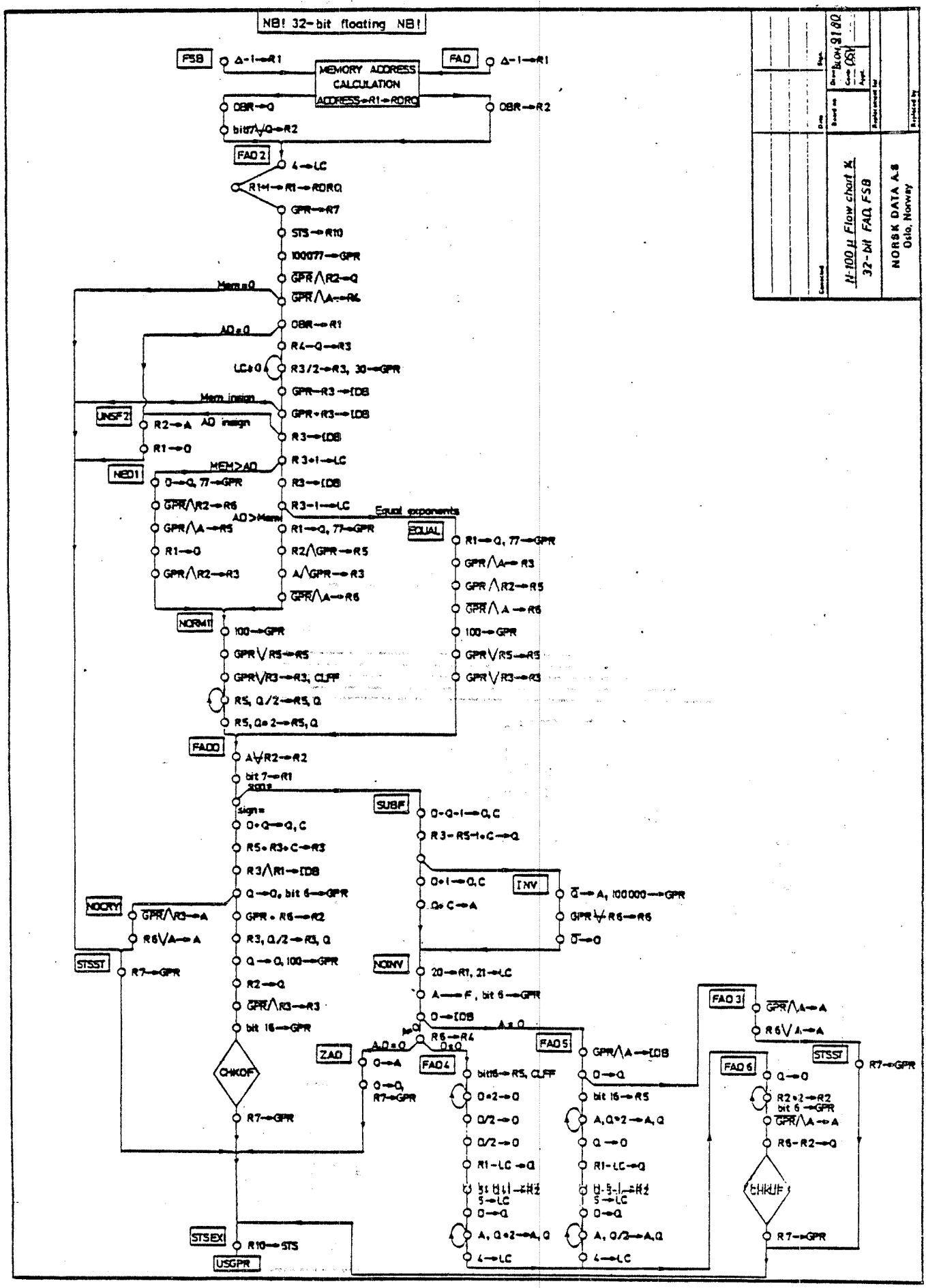

N-100 µ Flow chart 9
1504 xx PIOF,EXAM,DEPO,DXT
1500 xx TRA
1501 xx TRR
1433 xx SINTRAN III CORE MAP
NORSK DATA A.S
Oslo, Norway
Core map handling instructions for SINTRAN III
PION
VECT1
INPUT Fl.ch.6
IOF
IOF2 Fl.ch.5
ION
ION2 Fl.ch.5
Illegal
INIT Fl.ch.1
PMODX Fl.ch.1
PIOF
PION
EXAM
EXAM2
DEPO
DEPO2
IOXT
IOXT2 Fl.ch.4
REX
SEX
PON
POF
POF2
TRR
VECT3
RPANS
TRSTS
TRPCR
TRLMP
RLMP1
USDBR
RPCR1
TRIIE
RIIE1 Fl.ch.5
TRPID
RPID1 Fl.ch.5
TRCCL
RPIE1 Fl.ch.5
TRPIE
TRCIL
TRCIU
TR4
RTR4 ETC
TR13
TR14
TR16
TR17
TRECC
TRECC1
IOXT1 Fl.ch.4
APCG1
STAD
STA8
STZ8
LDA8
LDX8
LOB8
PLPCR
PLPC2
PRFCH
PRFCI
TRA
VECT2
APANS
AIR16
AIR17
AMM2
TASIS
TAPYL
APYL1
TAPGS
APGS1
TAIIC
AIIC1 Fl.ch.5
TAPID
API01 Fl.ch.5
TAPIE
APIE1
TAOPR
ACSR1
TACSR
TAPIM
TAALD
AALD1
TAPES
APES1
TAPGD
APGC1
TAPEA
USGPR Fl.ch.1
EXM01
EXM02
ALDVC
ALD-VALUE=0
Q→A Return
USDBR Fl.ch.1
SIS→Q
Q→SIS
A→Q
A→IOB
P→FETCH
SWAP→XMM
GRR→LDIR
A→XMM
Q→LMP Return
Q→LDPIL
Q→PID
Q→PIE
Q→IOB
Q→DERQ
Q→EXRQ
DBR→Q
DBR→X
DBR→A
DBR→I
DBR→GPR
Q→LDIR
Q→XXM (physical address)
I→IOB
D→DERQ
D→EXRQ
A→R7,DBR →GPR
121→Q
I→R7,DBR→GPR
SRCE→X→Q
R7→XMM (segment)
150122→LDIR
150121→LDIR
EXMOD→XMM
A→IOB
SIS→R6
SIS→R5
2→LC
A+2→R5 GPR→LDIR
R5→R5
R5→LDPIL
R6→LDPIL
A→XMM
MMR→Q
3773/Q→A
RMSK→R1
Q→A Return
MMR/Q→A
GPR→LDIR
177777→Q
160377→Q
153602/Q→A
PYL→Q
PIE→Q
OPR→Q
7→Q
PEA→A Return
PES→A Return
bit 15/Q→Q
bit 14/Q→Q
bit 13/Q→Q
bit 12/Q→Q
bit 11/Q→Q
bit 10/Q→Q
bit 9/Q→Q
bit 8/Q→Q
bit 7→MPID
bit PIL→A Return

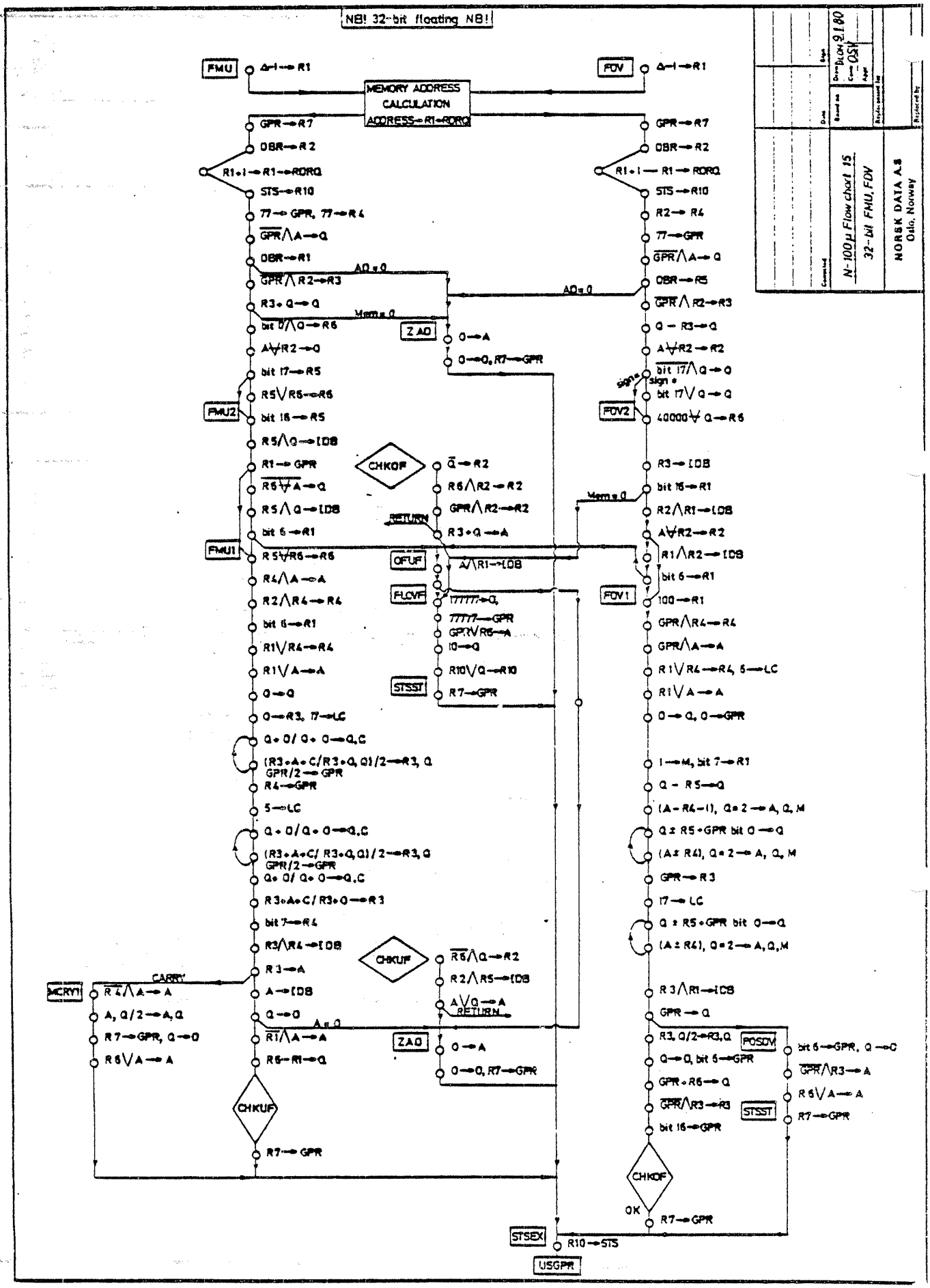


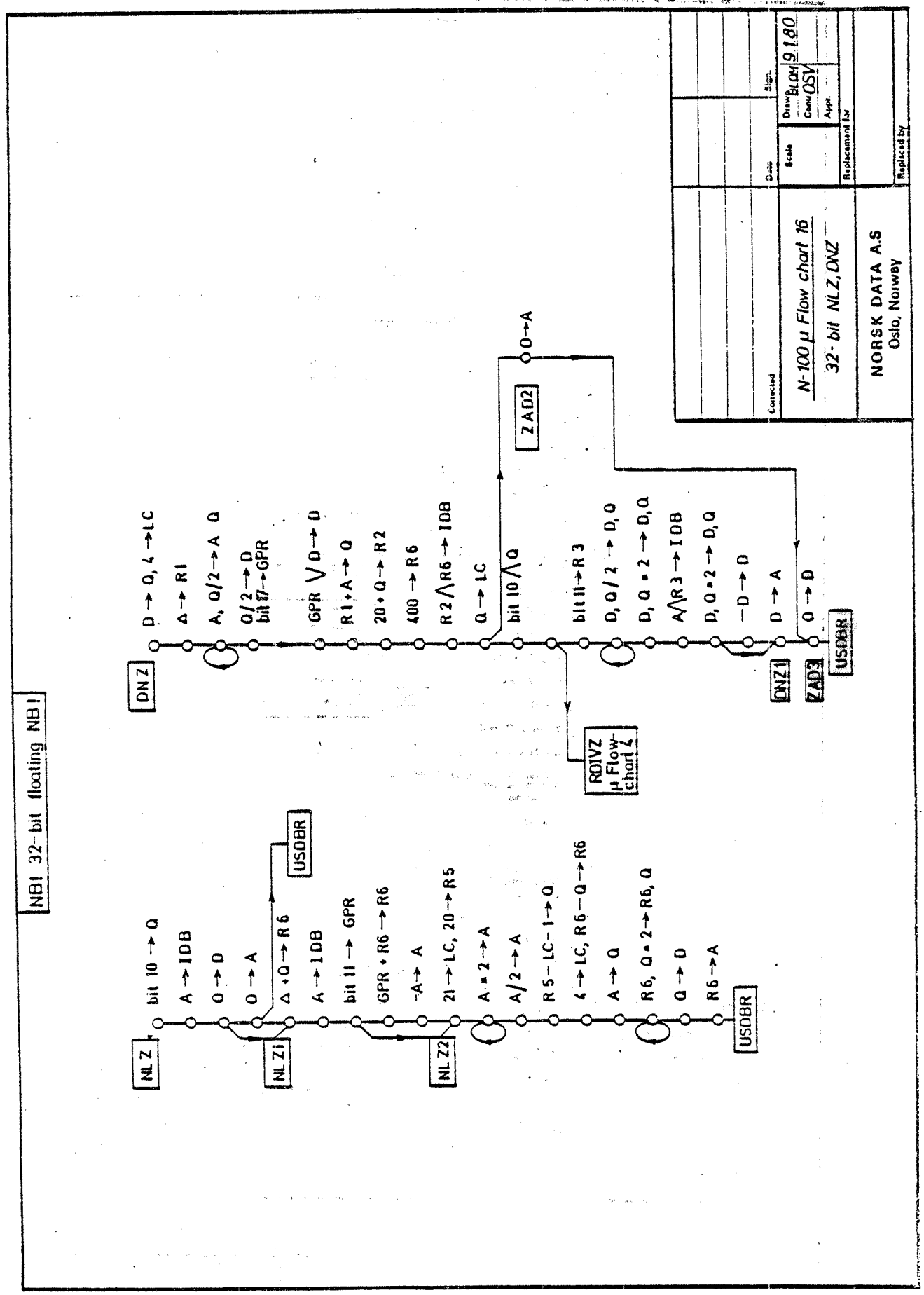

N-100 μ Flow chart II
MOVB, MOVBF 2
NORSK DATA A.S
Oslo, Norway


N-100 μ Flow chart 12
ROP instruction
ARG instruction
BOP instruction
Tegn. 21.12.79 OSV/ACR, BW


N-100 µ Flow chart 13
CPU self test, Memory test
MOPC-routines DSPLY & LABEL
NORSK DATA A.S
Oslo, Norway
SELFT
DSPLY
LABEL
ILLEG
RONLY
Return
Return to error loop point

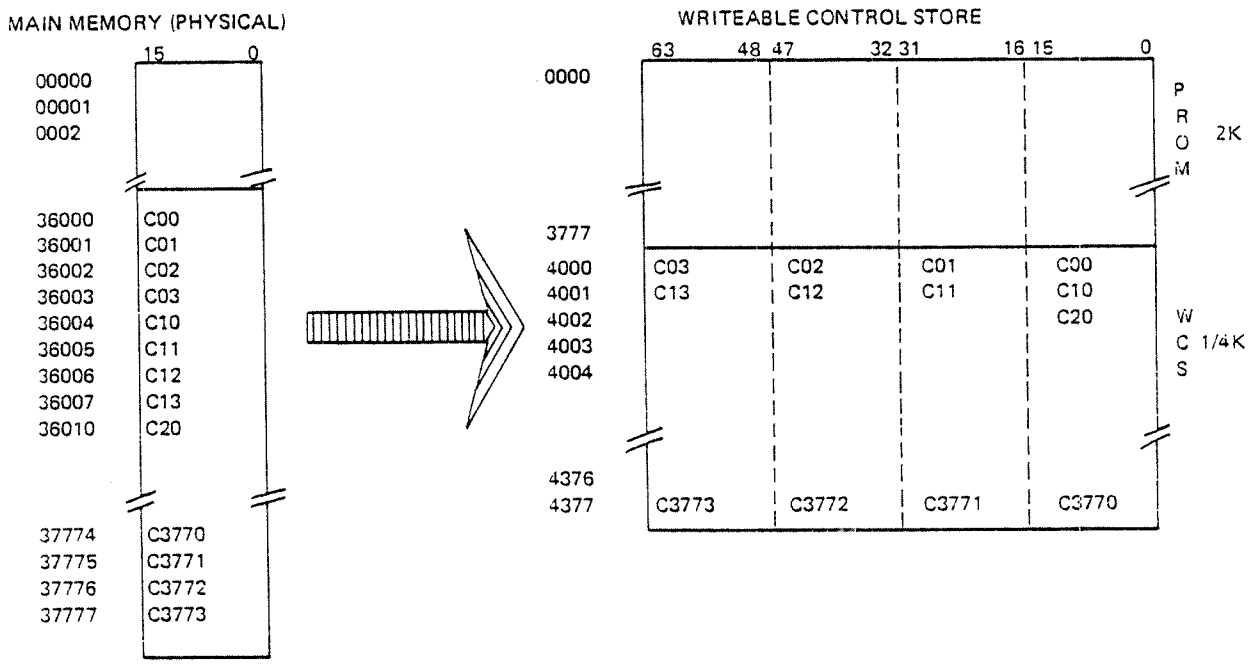

NB! 32-bit floating NB!
FSB
Δ-1→R1
MEMORY ADDRESS CALCULATION
ADDRESS→R1→RDRQ
FAD
Δ-1→R1
OBR→Q
OBR→R2
bit7∧/Q→R2
FAD 2
4→LC
R1=1→R1→RDRQ
GPR→R7
STS→R10
100077→GPR
GPR/\R2→Q
Mem≠0
GPR/\A→R6
AD≠0
OBR→R1
R4→Q→R3
LC≠0
R3/2→R3, 30→GPR
GPR→R3→DB
Mem insign
UNSF2
GPR→R3→DB
R2→A
AD insign
R3→DB
R1→Q
R3+1→LC
MEM>AD
NED1
0→Q, 77→GPR
R3→DB
GPR/\R2→R6
R3-1→LC
Equal exponents
AD>Mem
GPR/\A→R5
R1→Q, 77→GPR
EQUAL
R1→Q, 77→GPR
R1→Q
R2/\GPR→R5
GPR/\A→R3
GPR/\R2→R3
A/\GPR→R3
GPR/\R2→R5
GPR/\A→R6
GPR/\A→R6
NORM1
100→GPR
100→GPR
GPR\/R5→R5
GPR\/R5→R5
GPR\/R3→R3, CLFF
GPR\/R3→R3
R5, Q/2→R5, Q
R5, Q*2→R5, Q
FADD
A\/R2→R2
bit 7→R1
sign=
sign≠
SUBF
0-Q-1→Q, C
R3-R5-1+C→Q
0+Q→Q, C
R5+R3+C→R3
R3/\R1→DB
0+1→Q, C
INV
Q→A, 100000→GPR
Q→Q, bit 6→GPR
Q+C→A
GPR\/R6→R6
NOCRY
GPR/\R3→A
GPR+R6→R2
0→Q
R6\/A→A
R3, Q/2→R3, Q
STSST
R7→GPR
Q→Q, 100→GPR
NOINV
20→R1, 21→LC
R2→Q
A→F, bit 6→GPR
GPR/\R3→R3
0→DB
bit 16→GPR
R6→R4
A,Q≠0
A,Q=0
FAD 5
GPR/\A→DB
ZAD
FAD 4
CHKOF
0→A
0→Q
R7→GPR
bit15→R5, CLFF
0*2→Q
Q/2→Q
Q/2→Q
R1-LC→Q
bit 0-4→R2
5→LC
0→Q
A, Q*2→A, Q
4→LC
0→Q
bit 16→R5
A, Q*2→A, Q
Q→Q
R1-LC→Q
bit 0-5→R2
5→LC
0→Q
A, Q/2→A, Q
4→LC
FAD 3
GPR/\A→A
R6\/A→A
STSST
R7→GPR
FAD 6
Q→Q
R2+2→R2
bit 6→GPR
GPR/\A→A
R6-R2→Q
CHKUF
R7→GPR
STSEX
R10→STS
USGPR
N-100 μ Flow chart K
32-bit FAD FSB
NORSK DATA A.S
Oslo, Norway


NB! 32-bit floating NB!
FMU
Δ-1→R1
MEMORY ADDRESS CALCULATION
ADDRESS→R1←RORQ
GPR→R7
OBR→R2
R1+1→R1→RORQ
STS→R10
77→GPR, 77→R4
GPR/\A→Q
OBR→R1
AQ=0
GPR/\R2→R3
R3+Q→Q
Mem=0
bit 0/\Q→R6
Z AO
0→A
0→Q, R7→GPR
A\/R2→Q
bit 17→R5
R5\/R6→R6
FMU2
bit 18→R5
R5/\Q→DB
R1→GPR
CHKOF
Q→R2
R6/\R2→R2
GPR/\R2→R2
RETURN
R3+Q→A
R6\/A→Q
R5/\Q→DB
bit 6→R1
FMU1
R5\/R6→R6
QFUF
A/\R1→DB
R4/\A→A
FLOVF
R2/\R4→R4
177777→Q,
77777→GPR
GPR\/R6→A
bit 6→R1
10→Q
R1\/R4→R4
STSST
R10\/Q→R10
R1\/A→A
R7→GPR
0→Q
0→R3, 17→LC
Q+0/Q+0→Q,C
(R3+A+C/R3+Q, Q)/2→R3, Q
GPR/2→GPR
R4→GPR
5→LC
Q+0/Q+0→Q,C
(R3+A+C/ R3+Q, Q)/2→R3, Q
GPR/2→GPR
Q+0/ Q+0→Q,C
R3+A+C/ R3+0→R3
bit 7→R4
R3/\R4→DB
CHKUF
R6/\Q→R2
R3→A
R2/\R5→DB
CARRY
MCRY1
R4/\A→A
A\/Q→A
RETURN
A→DB
A, Q/2→A, Q
Q→0
A=0
R7→GPR, Q→0
R1/\A→A
ZAO
0→A
R6\/A→A
R6→R1→Q
0→Q, R7→GPR
CHKUF
R7→GPR
FOV
Δ-1→R1
GPR→R7
OBR→R2
R1+1→R1→RORQ
STS→R10
R2→R4
77→GPR
GPR/\A→Q
OBR→R5
AQ=0
GPR/\R2→R3
Q→R3→Q
A\/R2→R2
bit 17/\Q→Q
sign=
sign=
bit 17\/Q→Q
FOV2
40000\/Q→R6
R3→DB
bit 16→R1
Mem=0
R2/\R1→DB
A\/R2→R2
R1/\R2→DB
bit 6→R1
FOV1
100→R1
GPR/\R4→R4
GPR/\A→A
R1\/R4→R4, 6→LC
R1\/A→A
0→Q, 0→GPR
1→M, bit 7→R1
Q→R5→Q
(A-R4-1), Q+2→A, Q, M
Q±R5+GPR bit 0→Q
(A±R4), Q+2→A, Q, M
GPR→R3
17→LC
Q±R5+GPR bit 0→Q
(A±R4), Q+2→A, Q, M
R3/\R1→DB
GPR→Q
R3, Q/2→R3, Q
POSDV
Q→Q, bit 6→GPR
GPR+R6→Q
GPR/\R3→R3
STSST
bit 16→GPR
bit 6→GPR, Q→Q
GPR/\R3→A
R6\/A→A
R7→GPR
CHKOF
OK
R7→GPR
STSEX
R10→STS
USGPR
N-100μ Flow chart 15
32-bit FMU, FOV
NORSK DATA A.S
Oslo, Norway


NBI 32-bit floating NBI
NL Z
bit 10 → Q
A → IDB
0 → D
0 → A
NL Z1
Δ+Q → R6
USDBR
A → IDB
bit 11 → GPR
GPR + R6 → R6
-A → A
NL Z2
21 → LC, 20 → R5
A ∗ 2 → A
A/2 → A
R5 – LC-1 → Q
4 → LC, R6 – Q → R6
A → Q
R6, Q ∗ 2 → R6, Q
Q → D
R6 → A
USDBR
DN Z
D → Q, 4 → LC
Δ → R1
A, Q/2 → A Q
Q/2 → D
bit 17 → GPR
GPR ∨ D → D
R1 + A → Q
20 + Q → R2
400 → R6
R2 ∧ R6 → IDB
Q → LC
bit 10 ∧ Q
ZAD2
0 → A
RDIVZ
μ Flow-chart 4
bit 11 → R3
D, Q/2 → D, Q
D, Q ∗ 2 → D, Q
A ∨ R3 → IDB
D, Q ∗ 2 → D, Q
– D → D
D → A
0 → D
USDBR
DNZ1
ZAD3
Corrected
Date
Sign.
Scale
Drawn BLOM 9.1.80
Cont. OSV
Appr.
N-100 μ Flow chart 16
32-bit NLZ, DNZ
Replacement for
NORSK DATA A.S
Oslo, Norway
Replaced by

APPENDIX F

MAIN MEMORY/WCS CORRESPONDENCE



NORSK DATA A.S
P.O. Box 4, Lindeberg gård
Oslo 10, Norway

COMMENT AND EVALUATION SHEET

NORD-100 Microprogramming Description ND-06.018.01
January 1980

In order for this manual to develop to the point where it best suits your needs, we must have your comments, corrections, suggestions for additions, etc. Please write down your comments on this preaddressed form and mail it. Please be specific wherever possible.

FROM . . . . . .
. . . . . .
. . . . . .

– we make bits for the future

NORSK DATA A.S BOX 4 LINDEBERG GÅRD OSLO 10 NORWAY PHONE: 39 16 01 TELEX: 18661