GDC

I. Introduction:

The HP Nano Processor (NP) is a single chip, N-channel MOS, 8 bit parallel, control oriented central processing unit designed by the Loveland Instrument Division for internal control and interfacing of instruments.

The NP coupled with a program ROM forms the minimum nano processor control computer. The NP can directly address up to 2048 8 bit bytes of program memory and with simple block switching techniques up to 512K of 8 bit bytes.

All instructions and data are transferred in and out of the NP with the bidirectional 8 bit parallel data bus (DØ through D7).

The NP allows data transfers with up to 15 input and 15 output ports addressed by a 4 bit device select code and an I/O read/write control line.

The normal program may be interrupted by use of the interrupt request control line. This interrupt is a fully vectored interrupt with 256 possible vectors.

The NP can control external circuits and check their status through the use of the 7 direct control lines (DC0 through DC7).

All inputs and outputs are TTL compatabile. Each output will sink one standard power TTL load. Each input has an internal pull-up device.

The NP instruction set numbers 42 including data transfers, bit manipulation, magnitude comparisons, jump, and jump to subroutine.

HARDWARE STRUCTURE

The NP contains:

A. One 8 Bit Accumulator (ACC)

B. One Control Logic Unit (CLU)

C. One 1 Bit Extend Register (E)

D. Sixteen 8 Bit Storage Registers (RØ - R17)

E. One 8 Bit Magnitude Comparator (CMP)

F. Seven Bidirection Direct Control I/O Lines (DCØ-6)

G. One 11 Bit Program Counter (PC)

H. One 11 Bit Subroutine Stack Register (SSR)

I. One 11 Bit Interrupt Stack Register (ISR)

Accumulator

The 8 bit accumulator may be loaded from or output to the 8 bit data bus.

Control Logic Unit

The CLU is the heart of the NP. It provides the following functions:

1. Test, set or clear any bit of the accumulator or the extend register.
2. Set or clear any of the command flip flops.
3. Test any of the flag inputs.
4. Clear the accumulator.
5. Increment or decrement the accumulator in binary.
6. Increment or decrement the accumulator in decimal.

(Note: Two BCD coded digits are assumed and the output is two BCD coded digits and overflow.)

7. Complement accumulator (1's complement)

Extend Register

The 1 bit extend register is used to indicate overflow from or underflow to the accumulator, or it may be used as an internal flag.

Storage Registers

The sixteen 8 bit storage registers are for general data use. They may be recalled to the accumulator. They may be loaded from the accumulator or directly from the program ROM. RØ may be used for comparisons and indexing.

Magnitude Comparator

The magnitude comparator compares the 8 bits of the accumulator to the 8 bits of the RØ for greater than, less than or equal to.

Direct Control I/O Lines

The direct control I/O lines are 7 (DCØ-6) lines that may be used for output with set and clear functions on their controlling flip flops. The status of the output may be directly tested as inputs for feedback flags.

Program Counter

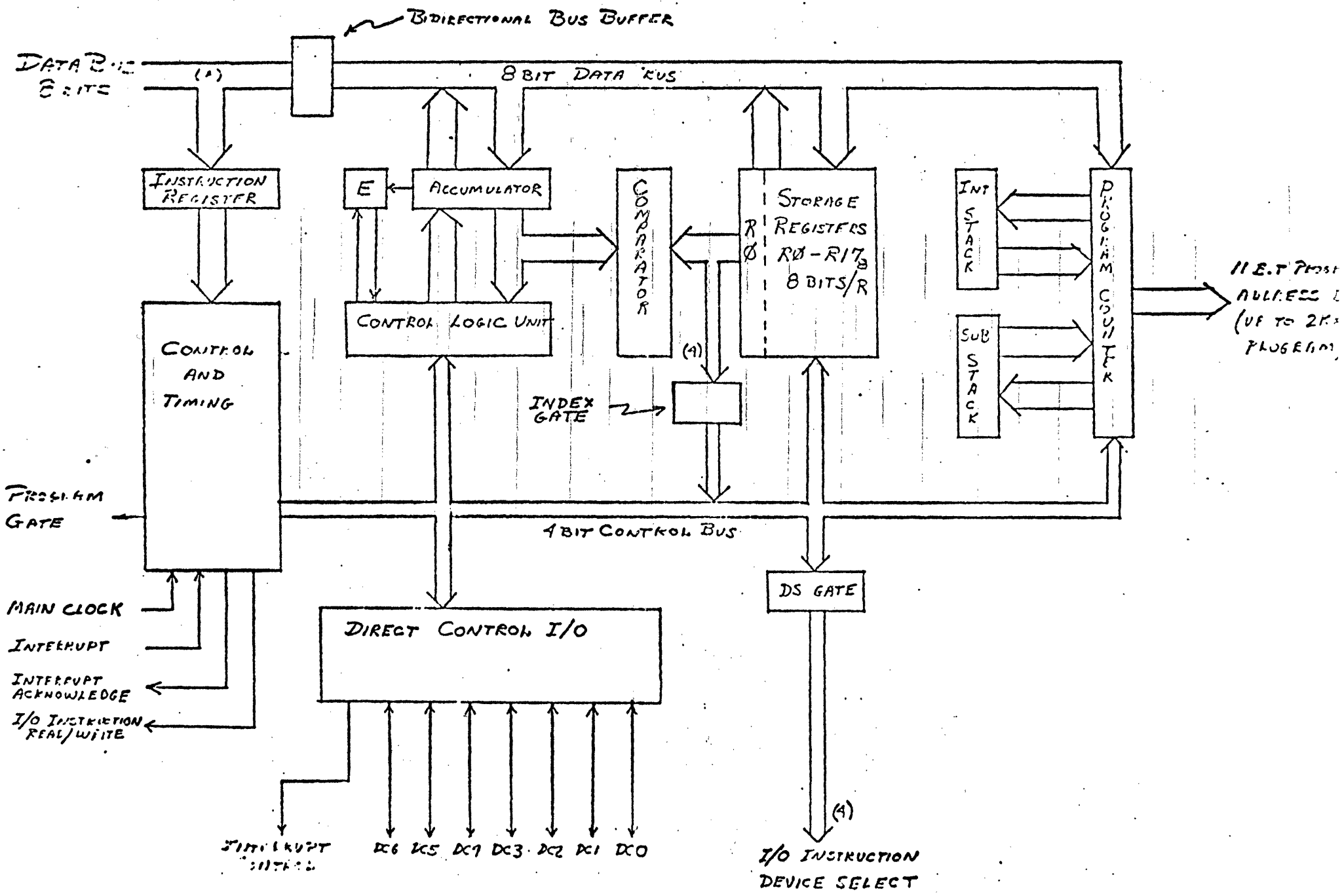
The 11 bit program counter provides direct addressing of the control program up to 2048 bytes.

Subroutine Stack Register

The 11 bit subroutine stack register provides for a single level of subroutining within the control program.

Interrupt Stack Register

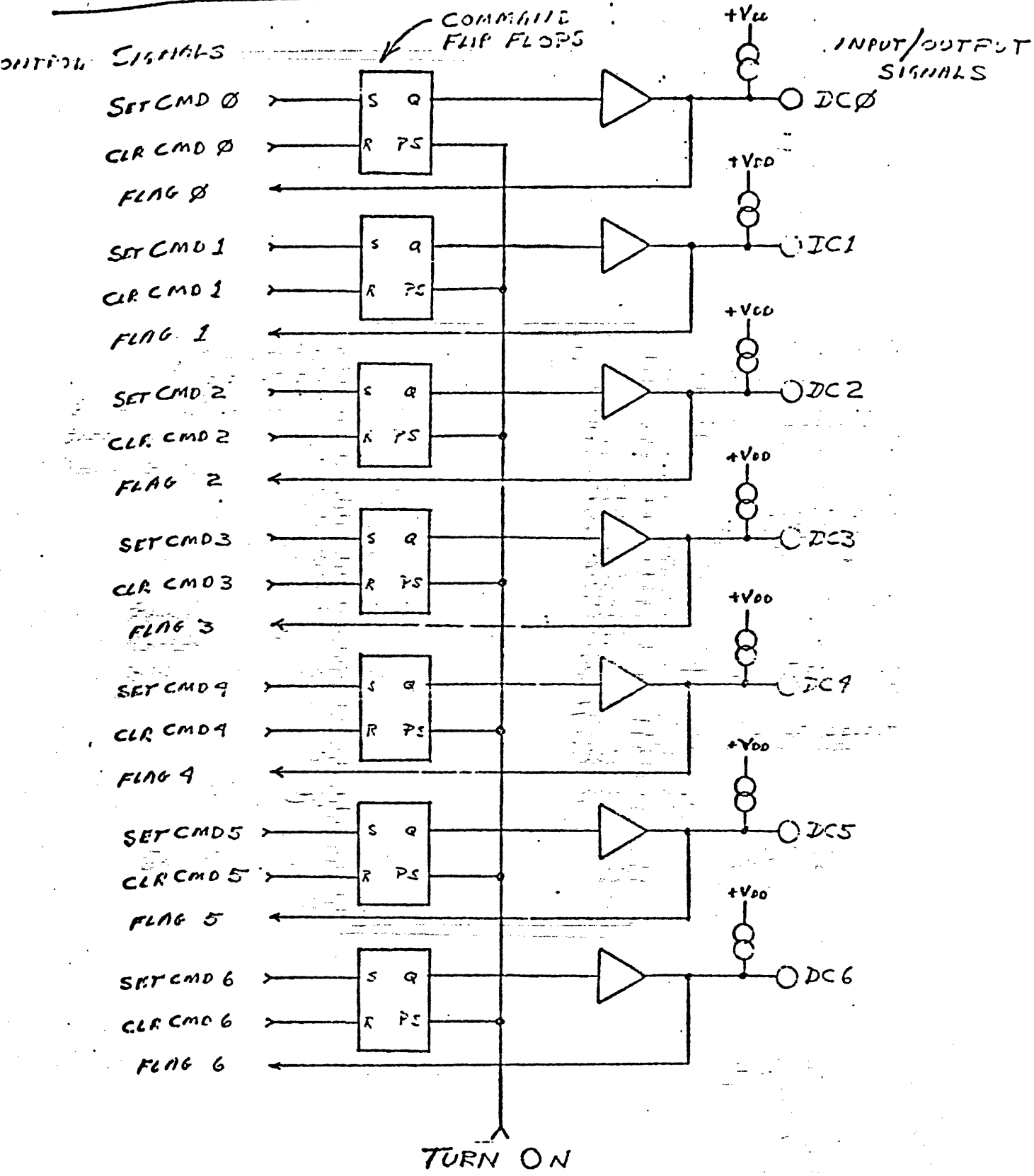
The 11 bit interrupt stack register provides for a single level of interruption.



NOTE: NOT ALL CONTROL LINES SHOWN.

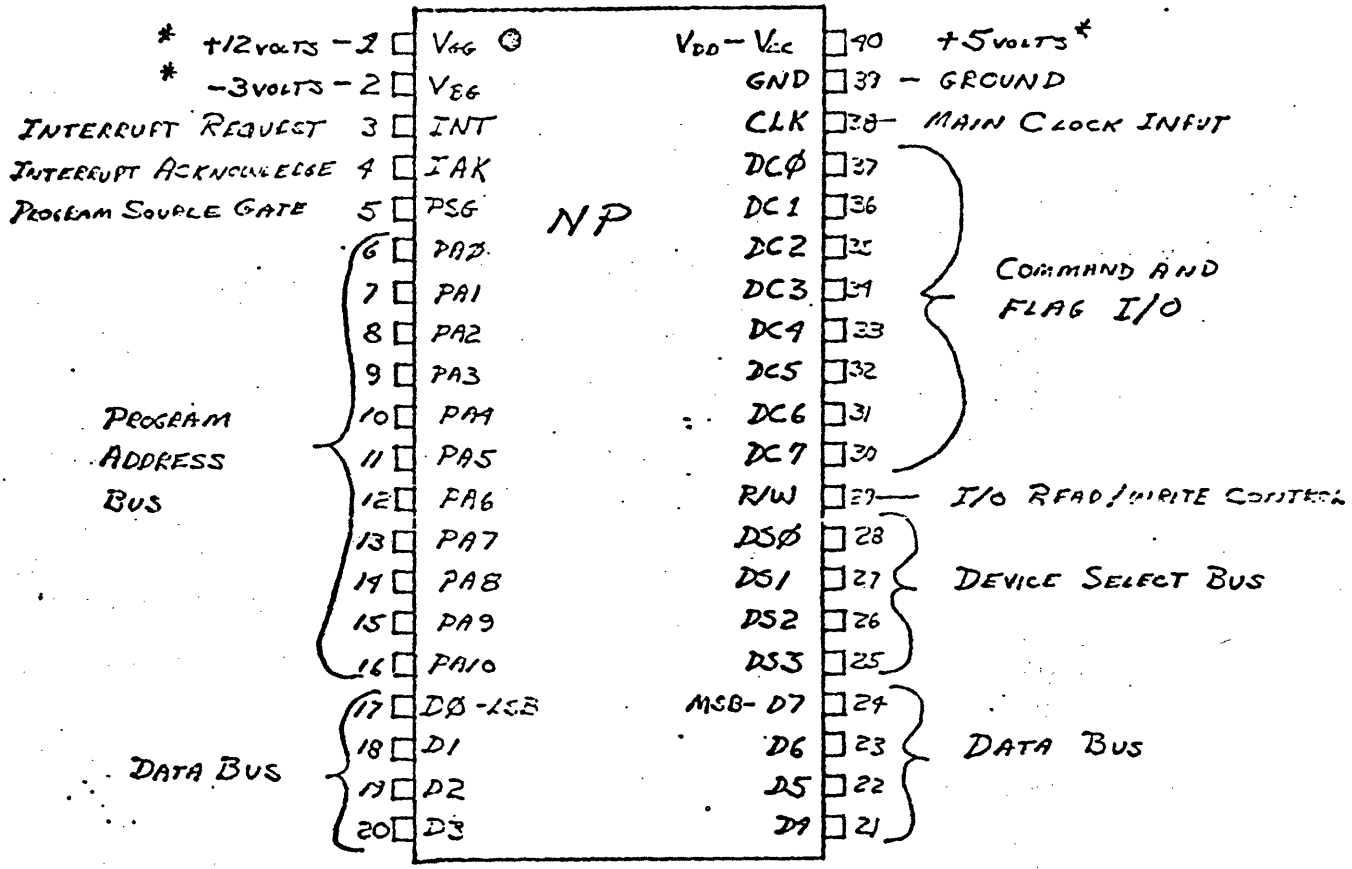
LARRY BOWLES 1/9/74

DIRECT CONTROL I/O STRUCTURE



LARRY BOWIE 9/5/74

NANO PROCESSOR PACKAGE



* ALL OPERATING VOLTAGE ARE NORMAL VOLTAGES ±5%

THIS DRAWING IS TO GIVE AN IDEA OF TERMINALS NOT THEIR CORRECT PIN NUMBER

LARRY BOWER

3/25/74

PROCESSOR TIMING

The NP is designed with a quasi static structure. The clock may be stopped in the low state with no loss of data.

The maximum clock rate is 5 mz. With this clock rate all instructions are executed in two clock periods or 400 ns.

To obtain a 400 ns cycle time the program ROM must have <70 ns access from address to output and <40 ns access from output enable to output. (A list of possible ROMS to be used with the NP is listed in Appendix A.)

INTRODUCTION TO THE NANOPROCESSOR INSTRUCTION SET

REGISTER ADDRESSING

The sixteen internal 8 bit registers may be directly addressed with LOAD (LDA), STORE (STA) and STORE ROM DATA (STR) instructions or indexed address may be used with LOAD INDEXED (LDI) and STORE INDEXED (STI).

The effective indexed address is the "or" function of the bottom 4 bits of the instruction with the bottom 4 bits of RØ(RØØ-RØ3). (I-I_3)

Example:

IØ-I3	1001
RØØ-RØ3	0101
Effective Register Address	1101

Note: This is an or function instead of an add, therefore no carry takes place.

PROGRAM ADDRESSING

For ease of discussion the program address (11 bits) will be looked at as a three bit page number (PA 1Ø-PA 8) and an 8 bit page offset (PA 7 - PA Ø)

In all instruction except jump and skip instructions, the program address is incremented. It is incremented once on one byte instructions and twice in two byte instructions.

In a JUMP (JMP) or JUMP TO SUBROUTINE (JSB) instruction the page number from the first byte and the page offset from the second byte of the instruction are loaded into the program counter during the execute phase.

In the JUMP INDIRECT INDEXED (JAI) and the JUMP INDIRECT INDEXED TO SUBROUTINE (JAS) instructions the page number is formed the same as an indexed register address (but only the bottom 3 bits are used) and the page offset is taken from the accumulator

CAUTIONS:

These two instructions allow great addressing power but they also have great dangers.

1. Due to the indexing structure a JAI instruction executed with RØ3 set will be executed as a JAS instruction.

2. Due to the subroutine return address storage system the address after a JAS instruction will not be executed upon return from the subroutine.

All branching in the NP is done with the skip instructions. The skip instruction causes two bytes of program to be skipped if the condition being tested is true.

Example:

	Program Address		Instruction
After the skip instruction	N		SBS 3 Skip if accumulator bit 3 is set
This instruction is executed if Bit 3 is zero →	N+1	constitutes the JMP instruction	JMP EXIT (Jump instructions require two bytes.)
	N+2		
This instruction is executed if Bit 3 is Set →	N+3		CBN 3 Clear accumulator bit 3

THE NANO PROCESSOR INSTRUCTION SET

The NP instruction set is divided into groups:

1. Accumulator group
2. Register transfer group
3. Input/output group
4. Comparator group
5. Program control group

INSTRUCTION LISTING FORMAT

SBS N

Ø	Ø	1	1	1	N	0

Instruction Mnemonic — Operand(s) — Operation Code — Operand Code

Description of Instruction

ACCUMULATOR GROUP

SBS N — 0 0 0 1 0 N

Skip on accumulator bit #N Set (1)

SBZ N — 0 0 1 1 0 N

Skip on accumulator bit #N zero (Ø)

SBN N — 0 0 1 0 0 N

Set accumulator bit #N

CBN N — 1 0 1 0 0 N

Clear accumulator bit #N

INB — 0 0 0 0 0 0 0 0

Increment accumulator as an 8 bit binary number

The extend register is set if overflow occurs

IND — 0 0 0 0 0 0 0 1

Increment accumulator as two BCD code decimal

numbers ( ) ( )

Carry between digits is automatically handled.

The extend register is set if overflow occurs.

DEB — 0 0 0 0 0 0 1 0

Decrement accumulator as an 8 bit binary number

The extend register is set if underflow occurs.

DED — 0 0 0 0 0 0 1 1

Decrement accumulator as two BCD coded decimal digits.

Borrow between digits is automatically handled.

The extend register is set if underflow occurs.

CLA		0 0 0 0 0 1 0 0
	Clear accumulator	
	Does not affect the extend register	
CMA		0 0 0 0 0 1 0 1
	Complement accumulator	
	The accumulator is treated as an 8 bit binary	
	number and one's complement is performed.	
LSA		0 0 0 0 0 1 1 0
	Left shift accumulator	
	1 bit shift with zero (Ø) fill	
	Does not affect extend register	
RSA		0 0 0 0 0 1 1 1
	Right shift accumulator	
	1 bit shift with zero (Ø) fill	
	Does not affect extend register	
SES		0 0 0 1 1 1 1 1
	Skip on extend register set (1)	
SEZ		0 0 1 1 1 1 1 1
	Skip on extend register Zero (Ø)	
LDR	ROM Data	1 1 0 a 1 1 1 1
	Load accumulator with ROM data	
	(ROM data is the second byte of this	
	instruction).	

REGISTER TRANSFER GROUP

LDA R — 0 1 1 0

Load accumulator with data from register #2

STA R — 0 1 1 1

Store accumulator at register #R.

LDI Z — 1 1 1 0

Load accumulator with data from register addressed by (Z)v(RØ) (See description of indexing

STI Z — 1 1 1 1

Store accumulator at register addressed by (z) v (RØ)

STR R, ROM Data — 1 1 0 1

ROM Data

Store ROM data at Register #R

ROM data is the second byte of this instruction

COMPARATOR GROUP

All comparisons are made based on RØ and the accumulator containing 8 bit unsigned binary numbers

---

SLT — 0 0 0 0 1 0 0 1

Skip on accumulator less than RØ.

SEQ — 0 0 0 0 1 0 1 0

Skip on accumulator equal to RØ.

SAZ — 0 0 0 0 1 0 1 1

Skip on accumulator equal to zero (Ø).

SLE — 0 0 0 0 1 1 0 0

Skip on accumulator less than or equal to RØ.

SGE — 0 0 0 0 1 1 0 1

Skip on accumulator greater than or equal to RØ.

SNE — 0 0 0 0 1 1 1 0

Skip on accumulator <u>not</u> equal to RØ.

SAN — 0 0 0 0 1 1 1 1

Skip on accumulator not equal to zero (Ø).

SGT — 0 0 0 0 1 0 0 0

> R0

INPUT/OUTPUT GROUP

INA DS — **0 1 0 0**

Input data from device #DS to accumulator

OTA DS — **0 1 0 1**

Output accumulator data to device #DS

OTR DS, ROM DATA — **1 1 0 0**

Output ROM data to device #DS

ROM data is the second byte of this instruction.

STC K — **0 0 1 0 1**

Set direct control

Bit #K

CLC K — **1 0 1 0 1**

Clear direct control

Bit #K

SFS J — **0 0 0 1 1**

Skip on direct control

Flag #J Set (1)

SFZ J — **0 0 1 1 1**

Skip on direct control flag #J zero (Ø)

RTI 10Ø10000

Return from interrupt

An unconditional jump to the location stored in the interrupt stack register is performed.

The interrupt control bit is not affected

RTE 10110001

Return from interrupt and enable interrupt

Same as RTI instruction except that the interrupt control bit is set allowing future interrupt.

NOP 01011111

NO Operation

JAI 10010

Jump indirect (through accumulator) indexed.

The page number is the indexed value (Z)V(RØ)

The page offset is the accumulator

An uncondition jump to the address formed from the page number and page offset.

JAS 10011

Jump indirect (through accumulator) indexed to subroutine.

Same as JAI with the addition that the location of the JAS instruction Plus 2 is stored in the subroutine stack register.

PROGRAM CONTROL GROUP

JMP ADDRESS — Page Number: 1 0 0 0 0; Page_Offset

The address is broken into two section page number and page offset.

The first byte contains operation code and page number.

The second byte contains the page offset.

An unconditional jump to the address is performed.

JSB ADDRESS — Page Number: 1 0 0 0 1; Page Offset

(See jump for address format)

An unconditional jump to the address is performed and the address of the next ROM location after the page offset is stored in the subroutine stack register.

Note: Since the subroutine stack register is a single level deep, subroutines <u>cannot</u> be nested.

RTS — 1 0 1 1 1 0 0 0

Return from subroutine

An unconditional jump to the location stored in the subroutine stack register is performed.

The location of the RTS instruction <u>Plus 2</u> is stored in the subroutine stack register, thus co-routine linkages may be performed.

INTERFACING THE NANOPROCESSOR

The interface of the NP is divided into five section:

1. Program Access
2. I/O Port
3. Direct Control Lines
4. Interrupt System
5. Power Supplies and Clock

PROGRAM ACCESS

The NP accesses its program through the use of the 11 program address lines (PAØ-1Ø) and the program and gate line.

When the program gate is high the program source should supply the program data referenced by the program address onto the data bus.

I/O PORTS

The NP can address up to 15 input and 15 output data ports through the use of its device select and I/O Read/Write lines.

The external devices may be numbered Ø through 16 in octal with 17 octal reserved for a "No I/O" indication.

DIRECT CONTROL LINES

The seven bidirectional direct control lines may be used in one of four modes for each line.

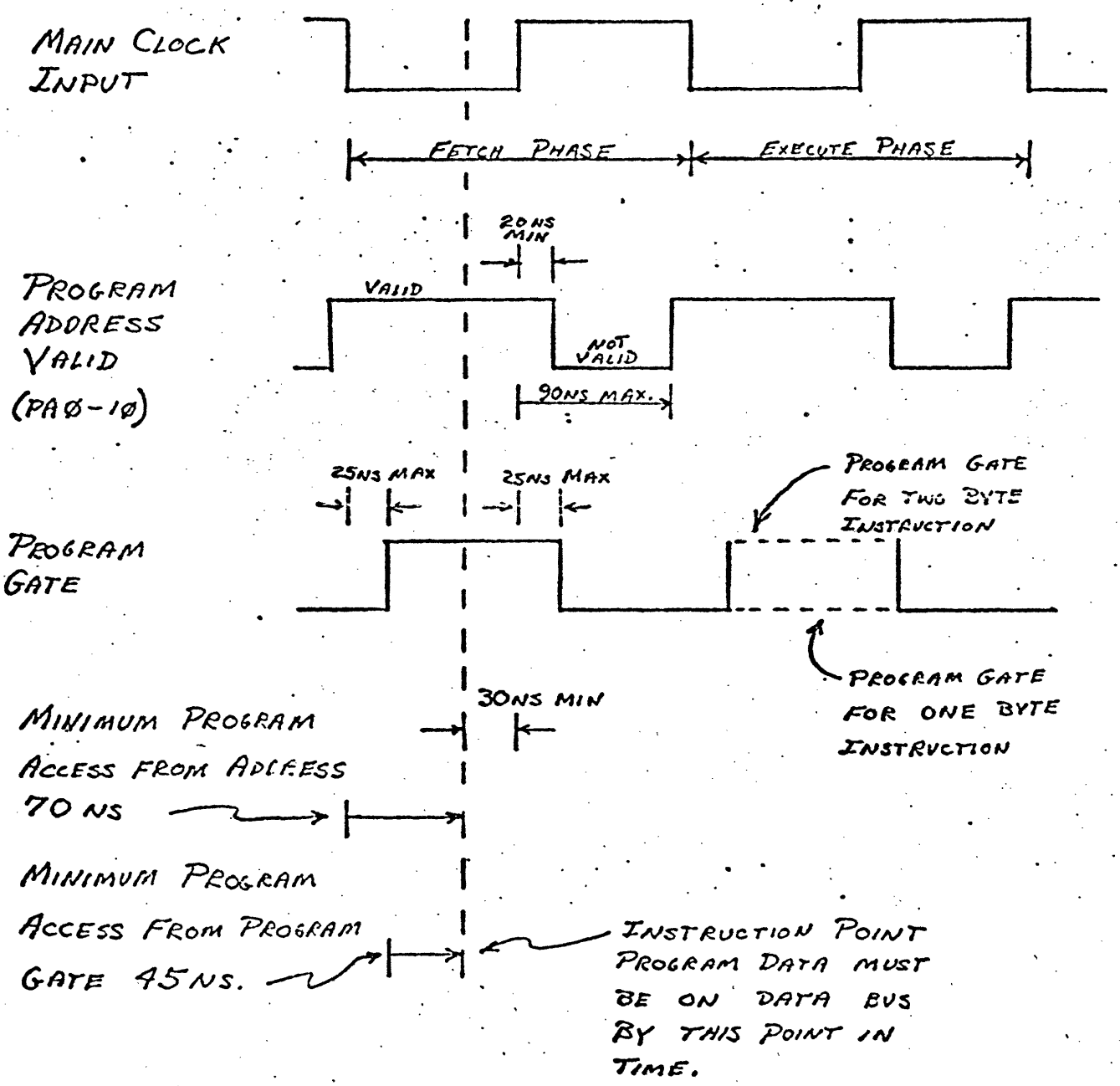
1. As a DC static output line with set/clear program control.
2. As an input flag (internal flip flop must be set - this is the turn-on condition) with direct testing by the program.
3. As a bidirectional control line.

   Example:

   The NP puts DC Line 2 low to signal an external device to start and the external device holds the line low until finished thus the NP (after setting DC lines again) can determine the end of the external devices cycle.
4. As an internal program flag with set/clear and direct testing by the program

PROGRAM ACCESS TIMING

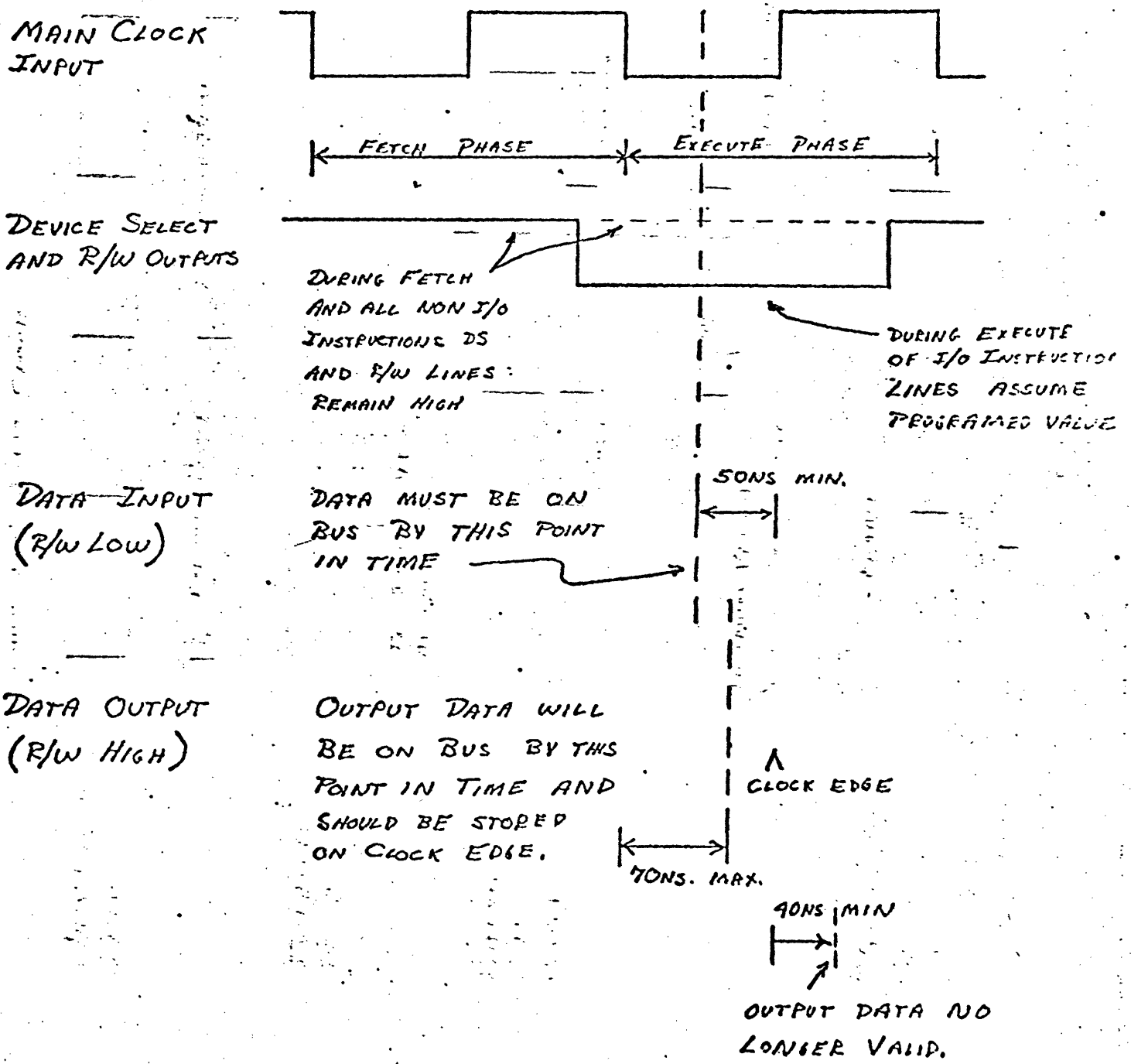
TIMING SHOWN FOR 5MHZ CLOCK



LARRY BOWER

I/O PORT TIMING

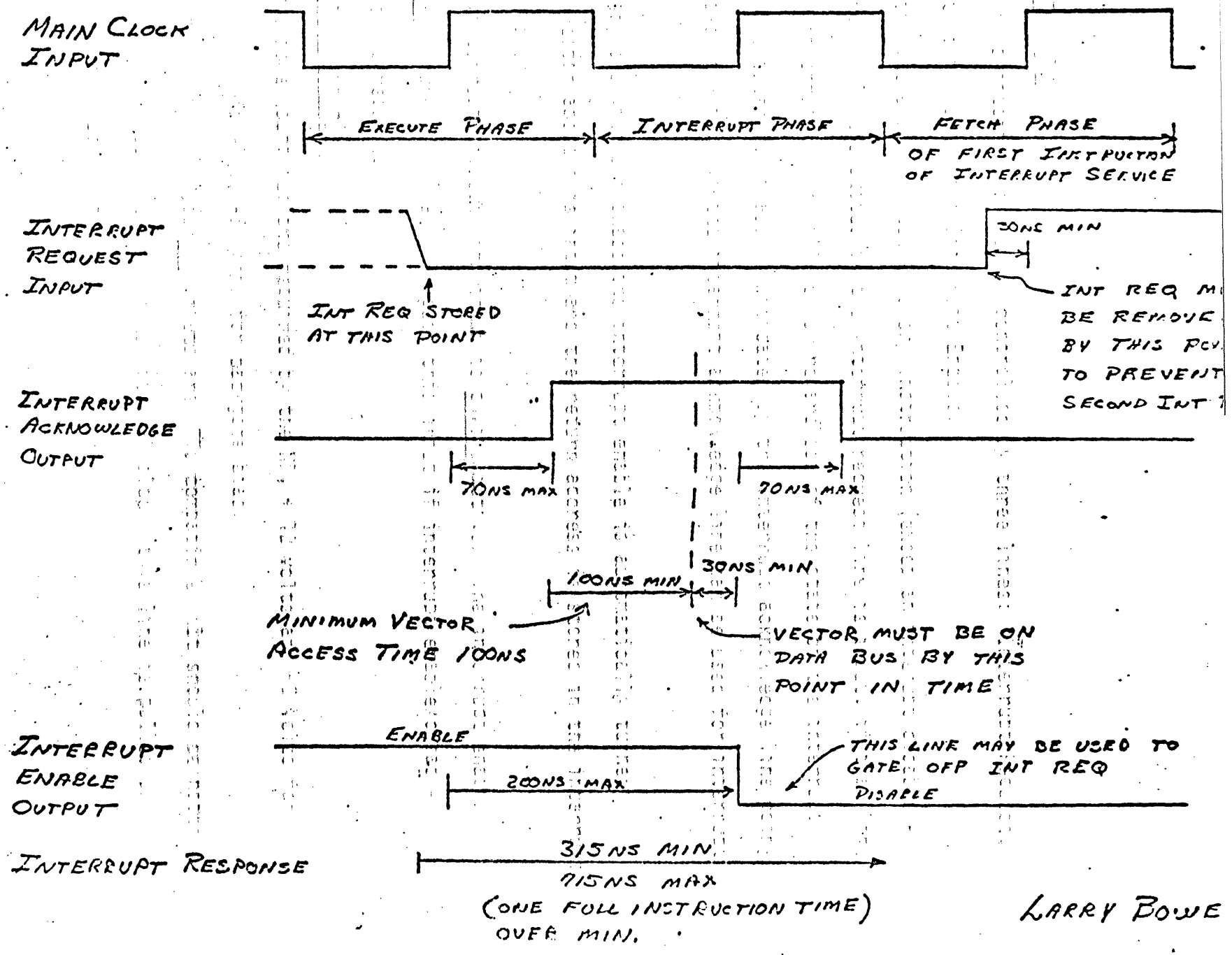
TIMING SHOWN FOR 5 MHZ CLOCK



LARRY BOWER

INTERRUPT SYSTEM TIMING

TIMING SHOWN FOR 5 MHz CLOCK



LARRY BOWE

INTERRUPT SYSTEM

The NP's interrupt system is controlled by three lines: Interrupt Request, Interrupt Acknowledge, and Interrupt Enable.

During the execute phase of every instruction (excep an interupt disable - clear control #7) the status of the interrupt request line is input, if that line is low an interrupt phase will follow regardless of the state of the interrupt enable. The interrupt phase is indicated by the interrupt acknowledge line going high. Daisy chaining of the interrupt acknowledge line can be used for interrupt priority.

During the interupt phase the interrupt enable is automatically turned off, the vector address is input and the return address is stored in the interrupt stack register.

The interrupt request line input is always active. The interrupt enable output may be used externally to gate this input if interrupt enable/disable capability is required.

POWER SUPPLY AND CLOCK

Three power supplies are required by the NP, + 12 volts and +5 Volts for the main logic and -2 to -5 volts for backgate bias.

The clock input is (as all inputs are) TTL compatible. It should be noted that to provide a fast clock edge the internal clock this line is pulled up with a current of approximately 3 ma.

REGISTER ADDRESSING

The sixteen internal 8 bit registers may be directly addressed with LOAD (LDA), STORE (STA) and STORE ROM DATA (STR) instructions or indexed address may be used with LOAD INDEXED (LDI) and STORE INDEXED (STI).

The effective indexed address is the "or" function of the bottom I-I_3) 4 bits of the instruction with the bottom 4 bits of RØ(RØØ-RØ3).

Example:

IØ-I3	1001
RØØ-RØ3	0101
Effective Register Address	1101

Note: This is an or function instead of an add, therefore no carry takes place.

PROGRAM ADDRESSING

For ease of discussion the program address (11 bits) will be looked at as a three bit page number (PA 1Ø-PA 8) and an 8 bit page offset (PA 7 - PA Ø)

In all instruction except jump and skip instructions, the program address is incremented. It is incremented once on one byte instructions and twice in two byte instructions.

In a JUMP (JMP) or JUMP TO SUBROUTINE (JSB) instruction the page number from the first byte and the page offset from the second byte of the instruction are loaded into the program counter during the execute phase.

In the JUMP INDIRECT INDEXED (JAI) and the JUMP INDIRECT INDEXED TO SUBROUTINE (JAS) instructions the page number is formed the same as an indexed register address (but only the bottom 3 bits are used) and the page offset is taken from the accumulator

CAUTIONS:

These two instructions allow great addressing power but they also have great dangers.

1. Due to the indexing structure a JAI instruction executed with RØ3 set will be executed as a JAS instruction.

2. Due to the subroutine return address storage system the address after a JAS instruction will not be executed upon return from the subroutine.

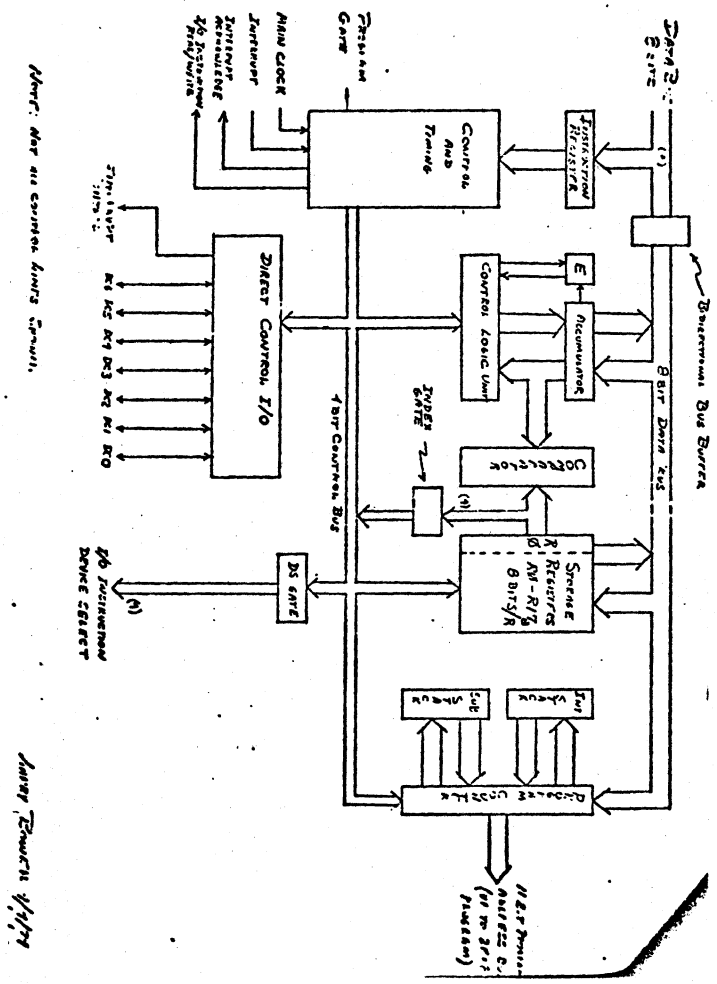
All branching in the NP is done with the skip instructions. The skip instruction causes two bytes of program to be skipped if the condition being tested is true.

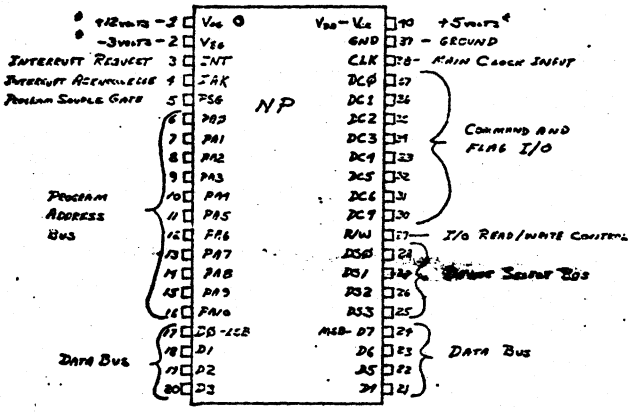
Example:

	Program Address	Instruction
After the skip instruction	N	SBS 3 Skip if accumulator bit 3 is set
This instruction is executed if Bit 3 is zero →	N+1	JMP EXIT (Jump instructions require two bytes
	N+2	
This instruction is executed if Bit 3 is Set →	N+3	CBN 3 Clear accumulator bit 3

INSTRUCTION LISTING FORMAT

Instruction Mnemonic	Operand(s)	Operation Code	Operand Code
SBS	N	ØØ111	N





INPUT/OUTPUT GROUP

INA DS 0100

Input data from device #DS to accumulator

OTA DS 0101

Output accumulator data to device #DS

OTR DS, ROM DATA 1100

Output ROM data to device #DS

ROM data is the second byte of this instruction.

STC K 00101

Set direct control

Bit #K

CLC K 10101

Clear direct control

Bit #K

SFS J 00011

Skip on direct control

Flag #J Set (1)

SFZ J 00111

Skip on direct control flag #J zero (Ø)

RTI 10Ø10000

Return from interrupt

An unconditional jump to the location stored in the interrupt stack register is performed.

The interrupt control bit is not affected

RTE 10110001

Return from interrupt and enable interrupt

Same as RTI instruction except that the interrupt control bit is set allowing future interrupt.

NOP 01011111

NO Operation

JAI 10010

Jump indirect (through accumulator) indexed.

The page number is the indexed value (Z)V(RØ)

The page offset is the accumulator

An uncondition jump to the address formed from the page number and page offset.

JAS 10011

Jump indirect (through accumulator) indexed to subroutine.

Same as JAI with the addition that the location of the JAS instruction Plus 2 is stored in the subroutine stack register

SBS N 00010 N

Skip on accumulator bit #N Set (1)

SBZ N 00110 N

Skip on accumulator bit #N zero (Ø)

SBN N 00100 N

Set accumulator bit #N

CBN N 10100 N

Clear accumulator bit #N

INB 00000000

Increment accumulator as an 8 bit binary number

The extend register is set if overflow occurs

IND 00000001

Increment accumulator as two BCD code decimal numbers ( ) ( )

Carry between digits is automatically handled.

The extend register is set if overflow occurs.

DEB 00000010

Decrement accumulator as an 8 bit binary number

The extend register is set if underflow occurs.

DED 00000011

Decrement accumulator as two BCD coded decimal digits.

Borrow between digits is automatically handled.

The extend register is set if underflow occurs.

CLA 00000100

Clear accumulator

Does not affect the extend register

CMA 00000101

Complement accumulator

The accumulator is treated as an 8 bit binary number and one's complement is performed.

LSA 00000110

Left shift accumulator

1 bit shift with zero (Ø) fill

Does not affect extend register

RSA 00000111

Right shift accumulator

1 bit shift with zero (Ø) fill

Does not affect extend register

SES 00011111

Skip on extend register set (1)

SEZ 00111111

Skip on extend register Zero (Ø)

LDR ROM Data 11011111

Load accumulator with ROM data

(ROM data is the second byte of this instruction)

REGISTER TRANSFER GROUP

LDA R 0110

Load accumulator with data from register #2

STA R 0111

Store accumulator at register #R.

LDI Z 1110

Load accumulator with data from register addressed by (Z)v(RØ) (See description of indexing

STI Z 1111

Store accumulator at register addressed by (z) v (RØ)

STR R, ROM Data 1101 ROM Data

Store ROM data at Register #R

ROM data is the second byte of this instruction

COMPARATOR GROUP

SLT 00001001

Skip on accumulator less than RØ.

SEQ 00001010

Skip on accumulator equal to RØ.

SAZ 00001011

Skip on accumulator equal to zero (Ø).

SLE 00001100

Skip on accumulator less than or equal to RØ.

SGE 00001101

Skip on accumulator greater than or equal to RØ.

SNE 00001110

Skip on accumulator not equal to RØ.

SAN 00001111

Skip on accumulator not equal to zero (Ø).

PROGRAM CONTROL GROUP

JMP ADDRESS 10000 Page Number / Page Offset

The address is broken into two section page number and page offset.

The first byte contains operation code and page number.

The second byte contains the page offset.

An unconditional jump to the address is performed.

JSB ADDRESS 10001 Page Number / Page Offset

(See jump for address format)

An unconditional jump to the address is performed and the address of the next ROM location after the page offset is stored in the subroutine stack register.

Note: Since the subroutine stack register is a single level deep, subroutines cannot be nested.

RTS 10111000

Return from subroutine

An unconditional jump to the location stored in the subroutine stack register is performed.

The location of the RTS instruction Plus 2 is stored in sub. stack reg.