MULTI
MICROMACHINE
DESCRIPTION

Copyright (C) 1976
Revision 1
11 March 1976
Nanodata Corporation
2457 Wehrle Drive
Williamsville, New York 14221
## CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>MULTI Micromachine Description</td>
<td>1</td>
</tr>
<tr>
<td>1.1</td>
<td>Resources Used by MULTI</td>
<td>1</td>
</tr>
<tr>
<td>1.2</td>
<td>Conventions</td>
<td>3</td>
</tr>
<tr>
<td>1.3</td>
<td>Instruction Set Summary and Index</td>
<td>5</td>
</tr>
<tr>
<td>2.0</td>
<td>MULTI Microinstruction Set</td>
<td>8</td>
</tr>
<tr>
<td>2.1</td>
<td>Control Store Operations</td>
<td>8</td>
</tr>
<tr>
<td>2.2</td>
<td>Arithmetic Operations</td>
<td>15</td>
</tr>
<tr>
<td>2.3</td>
<td>Shifting Operations</td>
<td>24</td>
</tr>
<tr>
<td>2.4</td>
<td>Main Store Operations</td>
<td>27</td>
</tr>
<tr>
<td>2.5</td>
<td>Branching Operations</td>
<td>29</td>
</tr>
<tr>
<td>2.6</td>
<td>Special Operations</td>
<td>35</td>
</tr>
<tr>
<td>2.7</td>
<td>System Support Operations</td>
<td>44</td>
</tr>
<tr>
<td>3.0</td>
<td>Microinstruction Set - Quick Reference</td>
<td>53</td>
</tr>
</tbody>
</table>
1.0 MULTI MICROMACHINE DESCRIPTION

The MULTI Micromachine is a multi purpose, microprogramming architecture. It was designed to be the base instruction set, whereby with the appropriate specialized extensions for any target architecture, the task of microprogramming an emulator can be easily and quickly accomplished. The architecture has 18/36-bit or 16/32-bit (Single/Double) data width, and 18 or 36-bit instruction width. The 80 instructions encompass the categories of Control Store operations, Arithmetic operations, Shifting operations, Main Store operations, Branching and testing operations, Special control and data operations, and System support operations. All of the basic operations in an emulation environment can be programmed with this instruction set. Extensions to this set are common in existing emulators but only necessary for speed and brevity at the micro level.

1.1 RESOURCES USED BY MULTI

1.1.1 Nanostore - MULTI uses 80 opcode words and 48 tail words of Nanostore. This leaves 48 opcode words and 80 tail words for any extension set which must fit within the basic two-page, 256 word, Nanostore block. This division works well because most extension sets for a specific emulator do not need a large number of new microinstructions, but they are complex and require a larger amount of tail space to complete the operation.
1.1.2 **Control Store and Main Store** - No Control Store or Main Store is necessary to support MULTI. They are the resources for the microprogrammer.

1.1.3 **Local Store Registers**

Six (6) registers (27-34) are used to supply information to specific MULTI instructions (e.g., addresses) and are therefore considered as shared between MULTI and the Micro level program.

Three (3) registers (35-37) are dedicated to the MULTI Nano level programs and are therefore not saved across interrupts.

Symbolic Name (Use) = QM-1 LS Register (octal)

<table>
<thead>
<tr>
<th>Symbolic Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>R.SYS (System Control)</td>
<td>27</td>
</tr>
<tr>
<td>R.MX (M.X. Address)</td>
<td>30</td>
</tr>
<tr>
<td>R.IX (C.S. Address)</td>
<td>31</td>
</tr>
<tr>
<td>R.IY (C.S. Array Addr.)</td>
<td>32</td>
</tr>
<tr>
<td>R.MPC (Micro P.C.)</td>
<td>33</td>
</tr>
<tr>
<td>R.ADR (C.S. Stack Addr.)</td>
<td>34</td>
</tr>
<tr>
<td>R.TMP (Nano Temp.)</td>
<td>35</td>
</tr>
<tr>
<td>R.SCR (Nano Scratch)</td>
<td>36</td>
</tr>
<tr>
<td>R.IR (Instr. Param.)</td>
<td>37</td>
</tr>
</tbody>
</table>
1.1.4 External Store and G-Store Registers

Eight (8) External Store registers (10-17) (i.e., Index Registers 0-7) are available to the user nanoprogrammer to support microinstruction extensions to MULTI. All other ES registers belong to MULTI and the Primary Control program. All G-Registers, except G11 (i.e., G0-G10) are the property of the user nanoprogrammer. G11 is a MULTI control and scratch register.

1.2 CONVENTIONS

The responsibilities of a nanoprogrammer choosing to extend this set include 1.) preservation of microcode interruptability after every microinstruction within an 8 microsecond period, and 2.) observation and maintenance of the MULTI conventions. The conventions of MULTI are as follows:

a. On entry to any microinstruction, the micro program counter contains the address of that microinstruction.

b. The COD bus contains the Control Store contents at MPC+1.

c. Local Store Register 31 contains the microinstruction parameters.
d. Both FCOD and FAIR contain 31.

e. The MPC will not be changed before the Trailing Edge of T period 2.

f. READ CS(MPC+2) will not be done before the Leading Edge of T period 3.

g. No assumptions should be made about the contents of NPC.
### INSTRUCTION SET SUMMARY AND INDEX

<table>
<thead>
<tr>
<th>CS OPERATIONS:</th>
<th>Page</th>
<th>ARITHMETICS:</th>
<th>Pag</th>
</tr>
</thead>
<tbody>
<tr>
<td>LD</td>
<td>8</td>
<td>LDI - Load Immediate</td>
<td>15</td>
</tr>
<tr>
<td>ST</td>
<td>8</td>
<td>LCI - Load Complement Immediate</td>
<td>15</td>
</tr>
<tr>
<td>LDX</td>
<td>9</td>
<td>ADI - Add Immediate</td>
<td>16</td>
</tr>
<tr>
<td>STX</td>
<td>9</td>
<td>SBI - Subtract Immediate</td>
<td>16</td>
</tr>
<tr>
<td>LDY</td>
<td>10</td>
<td>ADR - Add Register</td>
<td>17</td>
</tr>
<tr>
<td>STY</td>
<td>10</td>
<td>SBR - Subtract Register</td>
<td>17</td>
</tr>
<tr>
<td>LDD</td>
<td>11</td>
<td>ORR - OR Register</td>
<td>18</td>
</tr>
<tr>
<td>STD</td>
<td>11</td>
<td>ANR - AND Register</td>
<td>18</td>
</tr>
<tr>
<td>RAD</td>
<td>12</td>
<td>XOR - Exclusive Or Register</td>
<td>18</td>
</tr>
<tr>
<td>SW</td>
<td>12</td>
<td>NTR - Not Register</td>
<td>19</td>
</tr>
<tr>
<td>LDM</td>
<td>13</td>
<td>NGR - Negate Register</td>
<td>19</td>
</tr>
<tr>
<td>STM</td>
<td>13</td>
<td>MVR - Move Register</td>
<td>20</td>
</tr>
<tr>
<td>PULL</td>
<td>14</td>
<td>CPR - Compare Register</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ALUX - General ALU Function &amp; CPU Control Access</td>
<td>21</td>
</tr>
<tr>
<td>MS OPERATIONS:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LDMS</td>
<td>27</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STMS</td>
<td>27</td>
<td>LDN - Load Next CS Word</td>
<td>22</td>
</tr>
<tr>
<td>SWMS</td>
<td>27</td>
<td>ADN - Add Next CS Word</td>
<td>22</td>
</tr>
<tr>
<td>LDMSX</td>
<td>28</td>
<td>ONN - OR Next CS Word</td>
<td>22</td>
</tr>
<tr>
<td>STMSX</td>
<td>28</td>
<td>ANN - AND Next CS Word</td>
<td>23</td>
</tr>
<tr>
<td></td>
<td></td>
<td>XON - XOR Next CS Word</td>
<td>23</td>
</tr>
<tr>
<td>BRANCHING OPERATIONS:</td>
<td>Page</td>
<td></td>
<td></td>
</tr>
<tr>
<td>------------------------------------------</td>
<td>------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BPL - Branch Positive</td>
<td>29</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BNG - Branch Negative</td>
<td>29</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BZR - Branch Zero Register</td>
<td>30</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BNZ - Branch Non Zero</td>
<td>30</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BOS - Branch Ones Status</td>
<td>31</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BZS - Branch Zero Status</td>
<td>31</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BCT - Branch and Count</td>
<td>32</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B - Branch</td>
<td>32</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BALN - Branch &amp; Link Next</td>
<td>33</td>
<td></td>
<td></td>
</tr>
<tr>
<td>BALR - Branch &amp; Link Reg.</td>
<td>33</td>
<td></td>
<td></td>
</tr>
<tr>
<td>JUMP - Table Jump</td>
<td>34</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SHIFTING OPERATIONS:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(Immediate)</td>
<td></td>
</tr>
<tr>
<td>SRAI - Single Rt. Arith.</td>
<td>24</td>
</tr>
<tr>
<td>SLLI - Single Lt. Logical</td>
<td>24</td>
</tr>
<tr>
<td>SRLI - Single Rt. Logical</td>
<td>25</td>
</tr>
<tr>
<td>SRCI - Single Rt. Circular</td>
<td>25</td>
</tr>
<tr>
<td>SHIFT - Double General Shifting</td>
<td>26</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SPECIAL OPERATIONS:</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>EXTR - Split a Register</td>
<td>35</td>
</tr>
<tr>
<td>EXN - Execute Instruction in Next CS Word</td>
<td>35</td>
</tr>
<tr>
<td>EX - Execute Single</td>
<td>36</td>
</tr>
<tr>
<td>EXD - Execute Double</td>
<td>36</td>
</tr>
<tr>
<td>SBO - Set Bit to One</td>
<td>37</td>
</tr>
<tr>
<td>SBZ - Set Bit to Zero</td>
<td>37</td>
</tr>
<tr>
<td>TBO - Test Bit One</td>
<td>38</td>
</tr>
<tr>
<td>TBZ - Test Bit Zero</td>
<td>38</td>
</tr>
<tr>
<td>SETALU - Set ALU Mode</td>
<td>39</td>
</tr>
<tr>
<td>STAT - Set Status Reg.</td>
<td>39</td>
</tr>
<tr>
<td>ENQ - Enqueue Function</td>
<td>40</td>
</tr>
<tr>
<td>DEQ - Dequeue Function</td>
<td>42</td>
</tr>
<tr>
<td>SYSTEM - Start System State</td>
<td>43</td>
</tr>
<tr>
<td>SYSTEM SUPPORT OPERATIONS</td>
<td>Page</td>
</tr>
<tr>
<td>---------------------------</td>
<td>------</td>
</tr>
<tr>
<td>READS - Read Switches</td>
<td>44</td>
</tr>
<tr>
<td>AUX - Auxiliary Actions</td>
<td>44</td>
</tr>
<tr>
<td>SIDX - Set Nano Index</td>
<td>45</td>
</tr>
<tr>
<td>GENINT - Generate Interrupt</td>
<td>45</td>
</tr>
<tr>
<td>LDEI - Load ES. to LS.</td>
<td>46</td>
</tr>
<tr>
<td>STEI - Store LS. to ES.</td>
<td>46</td>
</tr>
<tr>
<td>XIO - Transmit I/O</td>
<td>47</td>
</tr>
<tr>
<td>RIO - Read I/O</td>
<td>47</td>
</tr>
<tr>
<td>HALT - Wait on Condition</td>
<td>48</td>
</tr>
<tr>
<td>LDMSA - Load MS Absolute</td>
<td>48</td>
</tr>
<tr>
<td>SAVE - Save Problem State</td>
<td>49</td>
</tr>
<tr>
<td>RESTORE - Restore Problem State</td>
<td>50</td>
</tr>
<tr>
<td>EXIT - Exit System State</td>
<td>51</td>
</tr>
<tr>
<td>STNS - Write Nanostore</td>
<td>52</td>
</tr>
</tbody>
</table>
2.0  MULTI MICROINSTRUCTION SET

2.1  CONTROL STORE OPERATIONS

Microinstruction:  LD  A,B

Condensed Description:  R(A) <= CS(R(B)); No Status.

Verbalized Description:  The contents of Control Store addressed by Register R(B) is loaded into Local Store Register R(A). Status in FIST is unchanged.  Timing = 7T.

Microinstruction:  ST  A,B

Condensed Description:  CS(R(B)) <= R(A); No Status.

Verbalized Description:  Local Store Register R(A) is stored at the Control Store location addressed by Register R(B). Status in FIST is unchanged.  Timing = 7T.
Microinstruction: \texttt{LDX A,B}

Condensed Description: \( R(A) \leftarrow CS(R(\text{R.IX}) + \text{SSSSBB}); \)
No Status.

Verbalized Description: The contents of Control Store addressed by the indexed value, Register \( R(\text{R.IX}) \) + sign extended "B", is loaded into the Local Store Register \( R(A) \). Status in FIST is unchanged. Timing = 8T.

Initial Assumptions: Local Store Register \( R,\text{IX} \) contains a Control Store address to be indexed by "B". (For LDX and STX both)

Microinstruction: \texttt{STX A,B}

Condensed Description: \( CS(R(\text{R.IX}) + \text{SSSSBB}) \leq R(A); \)
No Status.

Verbalized Description: The Local Store Register \( R(A) \) is stored at the Control Store location addressed by the indexed value, Register \( R(\text{R.IX}) \) + sign extended "B". Status in FIST is unchanged. Timing = 8T.
Microinstruction: LDY A,B

Condensed Description: R(R.IY) = R(R.IY) + SSSSBB,
                        then R(A) <= CS(R(R.IY)); No Status.

Verbalized Description: Local Store Register R(R.IY) is indexed
by the sign extended "B" parameter. Then Local Store Register R(A)
is loaded from the Control Store location addressed by R(R.IY).
The Status in FIST is not changed. Timing = 8T.

Initial Assumptions: Register R(R.IY) contains a Control
Store address to be indexed by "B" before fetching. (For both
LDY and STY)

Microinstruction: STY A,B

Condensed Description: R(R.IY) = R(R.IY) + SSSSBB,
                        then CS(R(R.IY)) <= R(A); No Status.

Verbalized Description: Local Store Register R(R.IY) is indexed
by the Sign extended "B" parameter. Then Local Store Register R(A)
is stored into the Control Store location addressed by R(R.IY).
The Status in FIST is unchanged. Timing = 8T.
Microinstruction: LDD A,B,V

Condensed Description: \( R(A) \leq CS(R(B) + V) \)
\( R(R.ADR) = R(B) + V; \) No Status.

Verbalized Description: Register \( R(R.ADR) \) receives the sum of Register \( R(B) + \) next CS location \( (V) \). Then the contents of Control Store addressed by \( R.ADR \) is loaded into Register \( (A) \). Status in FIST is unchanged. Timing = 10T.

Parameters: Register \( B \) cannot be an MPC register
and \( V \) is the second half of a double length microinstruction.
(Both LDD and STD)

Microinstruction: STD A,B,V

Condensed Description: \( CS(R(B) + V) \leq R(A) \)
\( R(R.ADR) = R(B) + V; \) No Status.

Verbalized Description: Register \( R(R.ADR) \) receives the sum of \( R(B) + \) next CS location \( (V) \). Then Register \( R(A) \) is stored at the Control Store location addressed by \( R.ADR \). Status in FIST is unchanged. Timing = 10T.
Microinstruction: \( \text{RAD} \ A, B, V \)

Condensed Description: \( R(R.\text{SCR}) = R(B) + V, \) then
\( R(A) <= R(A) + \text{CS}(R(R.\text{SCR})) , \)
then \( \text{CS}(R(R.\text{SCR})) <= R(A); \) ALU Status.

Verbalized Description: Local Store Register \( R(R.\text{SCR}) \) receives the sum of Local Store Register \( R(B) + \) the next CS location (V). Then Local Store Register \( R(A) \) receives the sum of \( R(A) + \) the contents of Control Store addressed by \( R.\text{SCR} \). The Control Store location addressed by \( R.\text{SCR} \) is then replaced by the new sum in \( R(A) \). ALU status in FIST reflects the resulting contents in \( R(A) \). Timing = 15T.

Parameters: B must be a Local Store Register which is not an MPC register. V is the second half of the microinstruction.

Result Conditions: The address computed into \( R.\text{SCR} \) is not available again for use.

Microinstruction: \( \text{SW} \ A, B \)

Condensed Description: \( R(A) <= \text{CS}(R(B)) \) and
\( \text{CS}(R(B)) <= R(A); \) No Status.

Verbalized Description: Swap Local Store Register \( R(A) \) with the contents of Control Store location addressed by Register \( R(B) \). Status in FIST is unchanged. Timing = 10T.
**Microinstruction:** LDM A,B

**Condensed Description:**
\[ R(B) \leftarrow CS(R(R.ADR)) \] thru
\[ R(B + A) \leftarrow CS(R(R.ADR) + A); \text{No Status.} \]

**Verbalized Description:** Load multiple Local Store Registers R(B) thru R(B+A), from A + 1 locations in Control Store, addressed from Register R(R.ADR) thru R(R.ADR) + A. Register R.ADR is incremented after every access. Status in FIST is unchanged.

Timing = 10 + 3(words)T.

**Result Conditions (for STM also):** \[ R(R.ADR) = R(R.ADR) + A + 1. \]

---

**Microinstruction:** STM A,B

**Condensed Description:**
\[ CS(R(R.ADR)) \leftarrow R(B) \] thru
\[ CS(R(R.ADR) + A) \leftarrow R(B + A); \text{No Status.} \]

**Verbalized Description:** Store multiple Local Store Registers R(B) thru R(B + A), into A + 1 locations in Control Store, addressed from Register R(R.ADR) thru R(R.ADR) + A. Register R.ADR is incremented after every access. Status in FIST is unchanged. Timing = 10 + 3(words)T.
Microinstruction: PULL A,B

Condensed Description: R(B + A) <= CS(R(R.ADR) - 1)
                    thru R(B) <= CS(R(R.ADR) - A - 1);
                    No Status.

Verbalized Description: Load multiple Local Store Registers
R(B + A) thru R(B), from A + 1 locations in Control Store, addressed
from Register (R(R.ADR) - 1) thru (R(R.ADR) - A - 1). Register
R.ADR is decremented once before each register to be loaded. Along
with STM, PULL can load a stack which STM has stored, with no
changes by the microprogrammer to R.ADR. The final address in
R.ADR indicates the last item which was placed into R(B). R.MPC
may be replaced as long as R.ADR is also loaded. This permits
transferring control while loading additional registers from the
stack in one operation. The status in FIST is unchanged.
Timing = 12 + 3(words)T.

Initial Assumptions: R(R.ADR) = top of stack + 1.

2.2 ARITHMETIC OPERATIONS

Microinstruction: \textit{LDI} A,B

Condensed Description: \( R(A) = 0000BB \); No Status.

Verbalized Description: The six-bit B parameter is loaded into Local Store Register R(A), right justified with zero fill. The status in FIST is unchanged. Timing = 5T.

Microinstruction: \textit{LCI} A,B

Condensed Description: \( R(A) = \text{.NOT. (0000BB)} \); No Status.

Verbalized Description: The six-bit B parameter is loaded into Local Store Register R(B), right justified with zero fill, and then it is complemented (ones-complement). The status in FIST is unchanged. Timing = 5T.
Microinstruction:       ADI   A,B

Condensed Description:  R(A) = R(A) + 0000BB;
                        ALU Status.

Verbalized Description: Local Store Register R(A) is incremented
                        by the right justified, zero filled, six-bit B parameter. The ALU
                        status in FIST reflects the final sum in R(A). Timing = 5T.

Microinstruction:       SBI   A,B

Condensed Description:  R(A) = R(A) - 0000BB;
                        ALU Status.

Verbalized Description: Local Store Register R(A) is decremented
                        by the right justified, zero filled, six-bit B parameter. The ALU
                        status in FIST reflects the final sum in R(A). Timing = 5T.
Microinstruction: ADR A,B

Condensed Description: \[ R(A) = R(A) + R(B); \]
ALU Status.

Verbalized Description: Local Store Register \( R(A) \) receives the sum of local store \( R(A) \) plus \( R(B) \). The ALU status in FIST reflects the final sum in \( R(A) \). Timing = 5T.

Microinstruction: SBR A,B

Condensed Description: \[ R(A) = R(A) - R(B); \]
ALU Status.

Verbalized Description: Local Store Register \( R(A) \) receives the difference of Local Store \( R(A) \) minus \( R(B) \). The ALU status in FIST reflects the final result in \( R(A) \). Timing = 5T.
Microinstruction: ORR A,B

Condensed Description: R(A) = R(A).OR.R(B); ALU Status.

Verbalized Description: Local Store Register R(A) receives the Logical OR of Local Store R(A).OR.R(B). The ALU status in FIST reflects the final result in R(A). Timing = 5T.

Microinstruction: ANR A,B

Condensed Description: R(A) = R(A).AND.R(B); ALU Status.

Verbalized Description: Local Store Register R(A) receives the Logical AND of Local Store R(A).AND.R(B). The ALU status in FIST reflects the final result in R(A). Timing = 5T.

Microinstruction: XOR A,B

Condensed Description: R(A) = R(A).XOR.R(B); ALU Status.

Verbalized Description: Local Store Register R(A) receives the Logical EXCLUSIVE OR of Local Store R(A).XOR.R(B). The ALU status in FIST reflects the final result in R(A). Timing = 5T.
Microinstruction: NTR A,B

Condensed Description: R(A) = .NOT.R(B); ALU Status.

Verbalized Description: Local Store Register R(A) receives the ones complement of Local Store R(B). ALU status reflects the result in R(A). Timing = 5T.

Microinstruction: NGR A,B

Condensed Description: R(A) = 0 - R(B); ALU Status.

Verbalized Description: Local Store Register R(A) receives the two's complement negation of Local Store R(B), i.e., the difference of 0 minus R(B). ALU status reflects the result in R(A). Timing = 5T.
Microinstruction: \textbf{MVR A,B}

Condensed Description: \(R(A) = R(B)\); No Status.

Verbalized Description: Local Store Register \(R(A)\) receives Local Store \(R(B)\). Status in FIST is unchanged. Timing = 5T.

Microinstruction: \textbf{CPR A,B}

Condensed Description: \(R(R.SCR) = R(A) - R(B)\);
\hspace{1cm} ALU Status.

Verbalized Description: Local Store Register \(R.SCR\) receives the difference of Local Store \(R(A)\) minus \(R(B)\). ALU status in FIST reflects this difference, i.e., an algebraic comparison with no residual affect to the microprogrammer's environment.

Result Conditions: \(R.SCR\) does not maintain this difference; use of \(SBR\) would hold the difference.
Timing = 5T.
Microinstruction: ALUX A,B,C,D,E

Condensed Description: \[ R(A) = R(B) \cdot \text{ALUOP}(C) \cdot R(E) \]
and \[ R(D) = \text{Control Register (E)}; \text{ALU Status}. \]

Verbalized Description: Local Store Register R(A) receives the result of ALU operation (C) on Local Store Registers R(B) and R(E) with initial carry-in reflecting the CARRY in FIST. If, the high bit of C is on, it requests decimal mode, and the (shifter output) 6's will be placed into R.ADR. ALU status reflects the result going into R(A). Also R(D) receives the CPU Control Register (E) (thru the Index ALU). If either result is not desired, causing R(A) or R(D) to point to R.SCR will eliminate that half of the operation. Timing = 10T.

Initial Assumptions: The carry bit in FIST is desired to affect the KALC control from the C parameter.

Restrictions: R(D) cannot be an MPC register. The two capabilities of this instruction are generally mutually exclusive, and R(E) is not the same as Control Register (E).
Microinstruction: \[ \text{LDN A,B,V} \]

Condensed Description: \[ R(A) = R(B) + V; \text{No Status.} \]

Verbalized Description: Local Store Register \( R(A) \) receives Local Store Register \( R(B) \) plus \( V \) (i.e., the next 18-bit word in Control Store). Status is unaffected. Timing = 8T.

Microinstruction: \[ \text{ADN A,B,V} \]

Condensed Description: \[ R(A) = R(B) + V; \text{ALU Status.} \]

Verbalized Description: Local Store \( R(A) \) receives the sum of Local Store \( R(B) \) plus \( V \) (the second half of the microinstruction). ALU Status in FIST reflects the sum. Timing = 8T.

Microinstruction: \[ \text{ORN A,B,V} \]

Condensed Description: \[ R(A) = R(B).OR.V; \text{ALU Status.} \]

Verbalized Description: Local Store \( R(A) \) receives the Logical OR of Local Store \( R(B).OR.V \) (the 2nd half of the microinstruction). ALU Status in FIST reflects this result. Timing = 8T.
Microinstruction: ANN A,B,V

Condensed Description: R(A) = R(B).AND.V; ALU Status.

Verbalized Description: Local Store R(A) receives the Logical AND of Local Store R(B).AND.V (i.e., the 2nd half of the microinstruction). ALU Status in FIST reflects the result. Timing = 8T.

Microinstruction: XON A,B,V

Condensed Description: R(A) = R(B).XOR.V; ALU Status.

Verbalized Description: Local Store R(A) receives the Logical EXCLUSIVE OR of Local Store R(B).XOR.V (i.e., the second half of the microinstruction). ALU Status in FIST reflects the result. Timing = 8T.
2.3 SHIFTING OPERATIONS

Microinstruction: SRAI A,B

Condensed Description: R(A) = R(A) shifted Right
Arithmetic B places; Sh Status.

Verbalized Description: Local Store R(A) is shifted Right
Arithmetic (i.e., sign fill), B places. Shifter status in FIST reflects the result. Timing = 5T.

Microinstruction: SLLI A,B

Condensed Description: R(A) = R(A) shifted Left
Logical B places, Sh Status.

Verbalized Description: Local Store R(A) is shifted Left
Logical (i.e., zero fill), B places. Shifter status in FIST reflects the result. Timing = 5T.
Microinstruction:  SRLI  A,B

Condensed Description:  R(A) = R(A) shifted Right
                        Logical B places; Sh Status.

Verbalized Description:  Local Store Register R(A) is shifted Right
                         Logical (zero fill) B places. Shifter Status in FIST reflects the
                         result. Timing = 5T.

Microinstruction:  SRCI  A,B

Condensed Description:  R(A) = R(A) shifted Right
                        Circular B places; Sh Status.

Verbalized Description:  Local Store Register R(A) is shifted Right
                         Circular (wrap around low end) B places. Shifter Status in FIST
                         reflects the result. Timing = 5T.
Microinstruction: \text{SHIFT A,B,C,D,E}

Condensed Description: \[ R(A-1)//R(A) = ALU(ALUOP(C),R(B-1))//R(B) \]
\begin{itemize}
  \item Shifted D-type, E-places;
  \item Sh & ALU Status.
\end{itemize}

Verbalized Description: Local Store Register pair $R(A-1)//R(A)$ receives Local Store Register pair $R(B-1)//R(B)$ shifted D-type and E-places. ALUOP(C) operates on $R(B-1)$ before the shift. ALU and Shifter status is saved in FIST reflecting the results (i.e., $KALC = C$, $KSHC = D$, $KSHA = E$). Timing = 11T.

Initial Assumptions: Low order bit of $R(B-1)$ is adjacent logically to the high order bit of $R(B)$. Carry bit in FIST will be used to SET (or CLEAR) COH to participate in shift.

Parameters: A and B may be equal if desired.
2.4  MAIN STORE OPERATIONS

Microinstruction:       LDMS    A,B

Condensed Description:  R(A)<= MS(R(B)); No Status.

Verbalized Description: Local Store R(A) receives the contents
of Main Store addressed by R(B). Status in FIST is unchanged.
Timing = 16T.

Operational Note (All MS instructions): Main Store addressing
may be absolute or relative as determined by global switch.

Microinstruction:       STMS    A,B

Condensed Description:  MS(R(B))<= R(A); No Status.

Verbalized Description: Local Store R(A) is stored at the location
in Main Store addressed by R(B). Status in FIST is unchanged.
Timing = 16T.

Microinstruction:       SWMS    A,B

Condensed Description:  R(A)<= MS(R(B)); No Status.

Verbalized Description: Local Store R(A) is swapped with the
contents of Main Store addressed by R(B). Status in FIST is
unchanged. Timing = 16T.
Microinstruction: LDMSX A,B

Condensed Description: R(A) <= MS(R(R.MX)) then
                          R(R.MX) = R(R.MX) + SSSSBB; No Status.

Verbalized Description: Local Store R(A) receives the contents of Main Store addressed by Local Store R(R.MX). Then register R.MX is indexed by the sign extended B parameter. Status in FIST is unchanged. Timing = 16T.

Initial Assumptions (for LDMSX and STMSX): R.MX has the Main Store address of the desired element. The high bit (5) of B is a sign (i.e., B address ranges from +31 to -32).

Microinstruction: STMSX A,B

Condensed Description: MS(R(R.MX)) <= R(A) then
                           R(R.MX) = R(R.MX) + SSSSBB; No Status.

Verbalized Description: Local Store R(A) is stored at the location in Main Store addressed by Local Store R(R.MX). Then register R.MX is indexed by the sign extended B parameter. Status in FIST is unchanged. Timing = 16T.
2.5 BRANCHING OPERATIONS

Microinstruction:  BPL  A,BR

Condensed Description:  \( R(R.MPC) = R(R.MPC) + SSSSBB \) if R(A) is positive,
else \( R(R.MPC) = R(R.MPC) + 1; \) No Status.

Verbalized Description:  If Local Store Register R(A) is positive (i.e., sign = 0) then increment \( R(R.MPC) \) by sign extended B parameter, else increment \( R(R.MPC) \) by one (i.e., remain in line).
Status in FIST is unchanged.  Timing = 11T (Branch), or 8T (No Branch).

Microinstruction:  BNG  A,BR

Condensed Description:  \( R(R.MPC) = R(R.MPC) + SSSSBB \)
if R(A) is negative,
else \( R(R.MPC) = R(R.MPC) + 1; \) No Status.

Verbalized Description:  If Local Store Register R(A) is negative (i.e., sign = 1) then increment \( R(R.MPC) \) by sign extended B parameter, else increment \( R(R.MPC) \) by one (i.e., remaining in line).
Status in FIST is unchanged.  Timing = 11T (Branch), 8T (No Branch).

Parameters (for all BR Branches):  -32 \leq B \leq +31.  Microassembler computes displacement from Microprogram Counter.
Microinstruction: BZR A,BR

Condensed Description: \[ R(R.MPC) = R(R.MPC) + SSSSBB \]
if R(A) is zero,
else \[ R(R.MPC) = R(R.MPC) + 1; \]
No Status.

Verbalized Description: If Local Store Register R(A) is zero (i.e., 18-bit zero) then increment R(R.MPC) by sign extended B parameter, else increment R(R.MPC) by one. Status in FIST is unchanged. Timing = 11T (Branch), or 8T (No Branch).

Microinstruction: BNZ A,BR

Condensed Description: \[ R(R.MPC) = R(R.MPC) + SSSSBB \]
if R(A) is non-zero,
else \[ R(R.MPC) = R(R.MPC) + 1; \]
No Status.

Verbalized Description: If Local Store Register R(A) is non-zero (i.e., any bit = 1) then increment R(R.MPC) by sign extended B parameter, else increment R(R.MPC) by one. Status in FIST is unchanged. Timing = 11T (Branch), or 8T (No Branch).
Microinstruction: BOS A,BR

Condensed Description: If A .AND. FIST is not zero,

then R(R.MPC) = R(R.MPC) + SSSSBB,

else R(R.MPC) = R(R.MPC) + 1; No Status.

Verbalized Description: If the status mask in the A parameter,
when ANDed with the real status in FIST has a non-zero result then
increment R(R.MPC) by the sign extended B parameter; else increment
R(R.MPC) by one. Status in FIST is unchanged. Timing = 11T
(Branch) or 8T (No Branch).

Parameters (for BOS and BZS): The A parameter is a 5-bit status
mask in FIST format, with set bits indicating which status bits
should be tested.

Microinstruction: BZS A,BR

Condensed Description: If A .AND. FIST is zero,

then R(R.MPC) = R(R.MPC) + SSSSBB,

else R(R.MPC) = R(R.MPC) + 1; No Status.

Verbalized Description: If the status mask (A parameter) when
ANDed with the status in FIST, has a zero result then increment
R(R.MPC) by sign extended B parameter; else increment R(R.MPC)
by one. Status in FIST is unchanged. Timing = 11T (Branch) or
8T (No Branch).
Microinstruction: \( BCT\ A, BR \)

Condensed Description: \( R(R.MPC) = R(R.MPC) + 1 \) if \( R(A) = 0 \) else, \( R(R.MPC) = R(R.MPC) + SSSSBB \)
and \( R(A) = R(A) - 1 \); No Status.

Verbalized Description: If Local Store Register \( R(A) \) is zero, then increment \( R(R.MPC) \) by one, else decrement \( R(A) \) by one and increment \( R(R.MPC) \) by sign extended \( B \) parameter. Status in FIST is unchanged. Timing = 12T (Branch), 9T (No Branch).

Microinstruction: \( B\ ABR \)

Condensed Description: \( R(R.MPC) = R(R.MPC) + SSAABB; \) No Status.

Verbalized Description: Unconditional relative branch is accomplished by incrementing \( R(R.MPC) \) by the 11-bit sign extended parameter \( A//B \). Status in FIST is unchanged. Timing = 8T.

Parameter: \( ABR \) is one 11-bit Micro Program Counter relative operand whose displacement is computed by the Micro-assembler.
Microinstruction: \textbf{BALR A,B}

Condensed Description: \[ R(A) = R(R.MPC) + 1 \]
\[ \text{then } R(R.MPC) = R(B); \text{ No Status.} \]

Verbalized Description: Branch and Link Register is accomplished by saving \( R(R.MPC) + 1 \) into Local Store Register \( R(A) \), then moving register \( R(B) \) into \( R.MPC \) and executing the instruction at \( CS(R(R.MPC)) \). Status in FIST is unchanged. Timing = 9T.

Microinstruction: \textbf{BALN A,B,V}

Condensed Description: \[ R(A) = R(R.MPC) + 2 \]
\[ \text{then } R(R.MPC) = R(B) + V; \text{ No Status.} \]

Verbalized Description: Branch and Link Next is accomplished by saving \( R(R.MPC) + 2 \) into Local Store Register \( R(A) \), then \( R(R.MPC) \) receives the sum of \( R(B) \) plus \( V \) (i.e., the 2nd half of instruction) and executing the instruction at \( CS(R(R.MPC)) \). Status in FIST is unchanged. Timing = 10T.

Parameters: Local Store Register \( R(B) \) cannot be an MPC register.
Microinstruction: JUMP A,B

Condensed Description: R(R.IR) = R(A).AND.0000BB,
then R(R.MPC) = R(R.MPC) + R(R.IR) + 1
then R(R.MPC) = CS(R(R.MPC)); No Status.

Verbalized Description: Jump via the case array which follows this instruction directly in Control Store. Mask a portion of Local Store register R(A) with the zero extended B parameter, into R.IR. Increment R(R.MPC) by R(R.IR) plus one. Load R(R.MPC) from Control Store addressed by the new R(R.MPC) and then execute the next instruction at CS(R(R.MPC)). Timing = 15T.

Parameters: The jump array of Control Store addresses are parameters directly in line with the JUMP instruction itself.

Example: JUMP R.CTL,7
                    address Ø
                    address 1
                    address 2
                    address 3
                    address 4
                    address 5
                    address 6
                    address 7
2.6 SPECIAL OPERATIONS

Microinstruction: EXTR A,B

Condensed Description: R(A) = High B bits of R(A) and
R(A + 1) = Low (18-B) bits of R(A);
ALU Status.

Verbalized Description: Extract the left most (high order) B bits
from Local Store Register R(A), and place them into R(A) right justi-
fied, with zero fill. The remaining (18-B) bits of R(A) are placed
into R(A + 1) left justified, with zero fill. ALU Status in FIST
reflects the resulting value in R(A). Timing = 5T.

Microinstruction: EXN A,B

Condensed Description: R(R.IR) = R(A) left shifted 6,
then R(R.IR) = R(R.IR).OR.R(B).OR.CS(R(R.MPC) + 1),
then execute instruction in R.IR; No Status.

Verbalized Description: Execute the instruction next in line, with
the parameters of CS(R(R.MPC)+1) ORed with Local Store Register R(A)
shifted left logical 6 places, and then ORed with register R(B).
The resulting instruction is placed into R.IR and executed. Status
in FIST is unchanged. Timing = 5T.

Parameters: Because R(A) and R(B) are ORed into R.IR
they may change the C field which is normally zero.

-35-
Microinstruction: EX A,B

Condensed Description: Execute (CS(R(B)) + R(A));
No Status.

Verbalized Description: Execute the instruction in Control Store addressed by Local Store Register R(B), which is added to R(A) (for parameters) and placed into R.IR. COD and R.MPC are still in line, i.e., R.MPC is not incremented and COD = CS(R(R.MPC) + 1). Status in FIST is unchanged. Timing: 9T.

Microinstruction: EXD A,B

Condensed Description: Execute Double (CS(R(B)) + R(A));
No Status.

Verbalized Description: Execute the double length instruction in Control Store addressed by Local Store Register R(B), which is added to R(A) (for parameters) and placed into R.IR. R.MPC is not incremented and COD = CS(R(B) + 1), for second half of the double word instruction. EXD requires one empty word in Control Store to accommodate 2 increments of R.MPC by the executed instruction. Status in FIST is unchanged. Timing = 9T.
Microinstruction:       SBO A,B

Condensed Description:   Set the 'B'\textsuperscript{th} bit R(A) one;
                         No Status.

Verbalized Description: Set the 'B'\textsuperscript{th} bit (i.e., 0 to 17 is
                         right to left) of Local Store Register R(A) to one. If B is
                         greater than 17 the instruction has no effect. Status in FIST
                         is unchanged. Timing = 7T.

Microinstruction:       SBZ A,B

Condensed Description:   Set the 'B'\textsuperscript{th} bit R(A) zero;
                         No Status.

Verbalized Description: Set the 'B'\textsuperscript{th} bit (i.e., 0 to 17 is right
                         to left) of Local Store Register R(A) to zero. If B is greater
                         than 17 the instruction has no effect. Status in FIST is unchanged.
                         Timing = 7T.
Microinstruction: TBO A,B

Condensed Description: If the 'B'\textsuperscript{th} bit of R(A) = 1
then \( R(R.MPC) = R(R.MPC) + 2 \),
else \( R(R.MPC) = R(R.MPC) + 1; \) No Status.

Verbalized Description: Skip the next instruction (i.e.,
\( R(R.MPC) = R(R.MPC) + 2 \)) if the 'B'\textsuperscript{th} bit of Local Store Register
R(A) is one, else drop thru (i.e., \( R(R.MPC) = R(R.MPC) + 1 \)).
Status in FIST is unchanged. Timing = 10T (skip), 7T (no skip).

Parameters (for TBO and TBZ): The 'B'\textsuperscript{th} bit is 0 thru 17, right to
left respectively.

Microinstruction: TBZ A,B

Condensed Description: If the 'B'\textsuperscript{th} bit of R(A) = 0
then \( R(R.MPC) = R(R.MPC) + 2 \),
else \( R(R.MPC) = R(R.MPC) + 1; \) No Status.

Verbalized Description: Skip the next instruction (i.e.,
\( R(R.MPC) = R(R.MPC) + 2 \)) if the B\textsuperscript{th} bit of Local Store Register
R(A) is zero, else increment R(R.MPC) by one. Status in FIST is
unchanged. Timing = 10T (skip), 7T (no skip).
Microinstruction: STAT A,B

Condensed Description: R(A) = old FIST, and FIST = B.

Verbalized Description: Local Store Register R(A) receives the old value of FIST, right justified with zero fill. Status in FIST is changed to the B parameter value. Timing = 5T.

---

Microinstruction: SETALU A,B

Condensed Description: R(A) = old FIDX, and 18-16 bit Mode bit = high bit of B.

Verbalized Description: Local Store Register R(A) receives the old value of FIDX, right justified with zero fill. The 18-16 bit Mode bit of FIDX will be set by the 5th (highest) bit of the B parameter, i.e. B = 1XXXXX yields 16 bit mode, B = 0XXXXX yields 18 bit mode. All other bits in FIDX are unchanged. Status in FIST is unchanged. Timing = 5T.
Microinstruction: ENQ A, BR

Condensed Description: Enqueue the QEL in CS(R(R.IX)) on to the Queue at CS(R(R.IY) + A), with special condition exit R(R.MPC) + SSSSBB; No Status.

Verbalized Description: Enqueue the QEL (Queue Element) in Control Store addressed by R.IX on to the Queue list identified by the QHDR (Queue Header) pointed at by R.IY (the QHDR Table address) plus the A parameter. Each QHDR has the following format:

<table>
<thead>
<tr>
<th>P.FIRST</th>
</tr>
</thead>
<tbody>
<tr>
<td>P.LAST</td>
</tr>
</tbody>
</table>

where P.FIRST points to the QEL at the Head of the Queue, and P.LAST to the last QEL on the chain, or is zero when the Queue chain is empty. Each QEL looks as follows:

<table>
<thead>
<tr>
<th>P.LINK</th>
</tr>
</thead>
<tbody>
<tr>
<td>P.ENTRY</td>
</tr>
</tbody>
</table>

where P.LINK 1.) points to the next QEL on the chain, or 2.) is the complement of the address of P.LINK itself if it is the last element on the chain, or 3.) is zero when not on a chain. P.ENTRY is a pointer to the data segment, or entry point for a code segment, governed by the Queue.
ENQ has three alternative procedures:

1.) when the QUEUE is inactive (P.LAST = 0). In this case ENQ sets P.FIRST = P.LAST = R(R.IX), and P.LINK = .NOT. R(R.IX). R(R.IY) receives P.ENTRY and control stays in line. The QEL is considered to be at the head of the Queue and the process may proceed on the data or code segment accessed.

2.) when the QUEUE is active (P.LAST ≠ 0). In this case the QEL is queued by setting P.LINK of QEL(P.LAST) = R(R.IX), then P.LAST = R(R.IX) and P.LINK of new QEL = .NOT. R(R.IX). This QEL is considered blocked and control is transferred to R(R.MPC) plus Sign extended B parameter.

3.) when the new QEL is already queued (P.LINK ≠ 0). In this case no queueing is done. R(R.IY) is set to its one's complement and control is transferred to R(R.MPC) + SSSSBB. This is considered an error condition.

Timing: Case 1 = 29T, Case 2 = 27T, Case 3 = 20T.
Microinstruction:       DEQ A,BR

Condensed Description:    Dequeue the QEL in CS(R.R.IX) from the
                          Queue at CS(R.R.IY + A), with special condition exit
                          R(R.MPC) + SSSSBB; No Status.

Verbalized Description:    Dequeue the QEL in Control Store
                          addressed by R.IX from the Queue list identified by the QHDR pointed
                          at by R.IY (the QHDR Table address) plus the A parameter.

DEQ has two alternative procedures:

1.) when the QEL to be dequeued is at the end of the Queue (P.LINK
    is negative).  In this case DEQ sets P.LINK = 0, deactivating the
    QEL, and P.LAST = 0, deactivating the Queue itself.  Control remains
    in line indicating no further processing necessary for this data or
    code segment.

2.) when the QEL to be dequeued is still linked (P.LINK is positive).
    In this case DEQ sets P.FIRST = P.LINK activating the next QEL in
    the chain, and sets P.LINK = Ø deactivating the old QEL.  Then
    DEQ sets R(R.IX) = P.FIRST and R(R.IY) receives the new P.ENTRY.
    Control transfers to R(R.MPC) + SSSSBB to begin accessing the data or
    code segment.

Timing:    Case 1 = 17T, Case 2 = 19T.
Microinstruction: SYSTEM B

Condensed Description: Initiates a logical interrupt, with B identifying the System Function.

Verbalized Description: SYSTEM initiates a logical interrupt, resulting in the transfer of control to a System Supervisor routine. Upon entering the Supervisor in Control Store partition zero, and Nano-Page zero, the state is changed to Supervisor State and a six word state save array will be saved in Control Store. This saves F-Registers FUSR, FACT, FIST, FIDX, FMPG, G11, three per word. Then it saves R.ADR, R.MPC, R.IY and R.IX in that order. The B parameter identifies the System Function and routine to be called. This parameter is passed to the System through Local Store Register R.ADR. Interrupt Disable is set through an AUX Action command before exit to the system's interrupt handler.

Timing = 42T.
2.7 SYSTEM SUPPORT OPERATIONS: ALL SYSTEM SUPPORT OPERATIONS ARE EXECUTED IN SUPERVISORY MODE ONLY

Microinstruction:  
AUX  A,B

Condensed Description:  Issue Auxiliary Action B, and 
R(A) = F-switches; No Status.

Verbalized Description:  Issue Auxiliary Action command B. 
Local Store register R(A) receives the six F-switches value, 
right justified with zero fill. If the AUX Action is an RMI load 
or load clock, RMI is set to one, and the value loaded is in Control 
Store addressed by R.ADR. If the AUX Action enables interrupts, 
interrupts will not be enabled until after the instruction 
following the AUX. Status in FIST is unchanged. Timing = 7T.

Microinstruction:  
READS  A

Condensed Description:  R(A) = F-switches; No Status.

Verbalized Description:  Local Store Register R(A) receives the 
six F-switches value, right justified with zero fill. If the 
B parameter has a value, this instruction will not assemble. 
Status in FIST is unchanged. Timing = 7T.
Microinstruction: SIDX A,B

Condensed Description: R(A) = old FIDX, and FIDX = B; No Status.

Verbalized Description: Local Store Register R(A) receives the old FIDX value, right justified with zero fill, and FIDX is set to the B parameter. The B parameter is assumed to be in FIDX form to set the Nanostore page index, system state, read only mode, and ALU width controls. The next instruction will execute in the new state. Status in FIST is unchanged. Timing = 5T.

Microinstruction: GENINT B

Condensed Description: Generates (or clears) the 'B' th physical interrupt; No Status.

Verbalized Description: The 'B' th physical interrupt level (i.e., bits 0 thru 4 of B) is generated or cleared (i.e., bit 5 = 0 (generate) or 1 (clear)). If the interrupt is generated it will be serviced immediately following the GENINT if no higher priority interrupt is Pending and Enabled. Status in FIST is unchanged. Timing = 6T.
Microinstruction: LDEI A,B

Condensed Description: \( R(A) = ES(B) \); No Status.

Verbalized Description: Local Store Register \( R(A) \) receives External Store Register \( ES(B) \). Status in FIST is unchanged. Timing = 5T.

Microinstruction: STEI A,B

Condensed Description: \( ES(B) = R(A) \); No Status.

Verbalized Description: Local Store Register \( R(A) \) is stored into External Store Register \( ES(B) \). Status in FIST is unchanged. Timing = 5T.

Microinstruction: PCHECK

Condensed Description: Program Check.

Verbalized Description: This is not a legal operation code but execution of any opcode in the form of 010XXX will cause a program check which transfers to Nanostore location ONE.
Microinstruction: XIO A,B

Condensed Description: Transmit data in R(A) and Function B to port(R.ADR).

Verbalized Description: Local Store Register R(A) data is transmitted along with the I/O Function code B through the port and to the device indicated in Local Store R(R.ADR) as follows: R.ADR bits 6 thru 8 = port number, and bits 0 thru 5 = device number. Status in FIST is unchanged. Timing = 8T.

Microinstruction: RIO A,B

Condensed Description: Receive data into R(A) and Device ID into R(B) from port(R.ADR).

Verbalized Description: Local Store Register R(A) receives the data from the port indicated in Local Store R(R.ADR) as follows: R.ADR bits 6 thru 8 = port number. Local Store Register R(B) receives the Device Identification number right justified with zero fill. Status in FIST is unchanged. Timing = 8T.
Microinstruction: HALT B

Condensed Description: Conditional System Halt based on F-Switches (B).

Verbalized Description: The System will halt if the F-Switch addressed by B is on (up), and will continue if and when it is turned off (down). The bits in B are therefore a mask against the value in the current F-Switch setting, causing the system to hang in this microinstruction until a zero results from B.AND.F-Switches. Status in FIST is unchanged. Timing = 10T minimum.

Microinstruction: LDMSA A,B

Condensed Description: Load R(A) <= MS(R(R.MX)) absolute and then R(B) = R(B) + R(A) and R(R.MX) = R(R.MX) + 1.

Verbalized Description: Load from absolute Main Store addressed by R(R.MX) into Local Store Register R(A) and increment R.MX by one. Then add R(A) into the check sum in R(B). Status in FIST is unchanged. Timing = 16T.
Microinstruction: SAVE B,A

Condensed Description: Saves Index Registers and G-Registers at CS(R(A)), in B + 5 locations.

Verbalized Description: Saves the active process Index Registers from External Store ES(8) through ES (8 + B), in Control Store addressed by Local Store Register R(A), incrementing R(A) once for each register stored. Then SAVE saves the G Registers (F20 thru F29), three per word, in the following Control Store locations. R(A) ends pointing one past the end of the save array. Timing = 52T (for B = 9 Index Registers).

Parameters: B must be greater than zero.