LOGICAL DESIGN FOR FAST SERIAL COMPUTER

TECHNICAL REPORT NO. ESD-TR-65-81

JULY 1965

H. B. Enderton

Prepared for

DIRECTORATE OF COMPUTERS
ELECTRONIC SYSTEMS DIVISION
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE

L. G. Hanscom Field, Bedford, Massachusetts

Project 508
Prepared by
THE MITRE CORPORATION
Bedford, Massachusetts
Contract AF19(628)-2390

Qualified requesters may obtain copies from DDC. Orders will be expedited if placed through the librarian or other person designated to request documents from DDC.

When US Government drawings, specifications, or other data are used for any purpose other than a definitely related government procurement operation, the government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise, as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

Do not return this copy. Retain or destroy.
LOGICAL DESIGN FOR FAST SERIAL COMPUTER

TECHNICAL REPORT NO. ESD-TR-65-81

JULY 1965

H. B. Enderton

Prepared for
DIRECTORATE OF COMPUTERS
ELECTRONIC SYSTEMS DIVISION
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE
L. G. Hanscom Field, Bedford, Massachusetts

Project 508
Prepared by
THE MITRE CORPORATION
Bedford, Massachusetts
Contract AF19(628)-2390
LOGICAL DESIGN FOR FAST SERIAL COMPUTER

ABSTRACT

The detailed logical design is given for a serial digital computer using a 0.5-microsecond magnetic memory, 100-mc logical circuits, and one-word delay lines. The computer performs most instructions in 1 microsecond. A list is given of the amount of hardware used.

REVIEW AND APPROVAL

This technical documentary report has been reviewed and is approved.

James A. Mellin
Major, USAF
Chief, Computer Division
CONTENTS

SECTION I - INTRODUCTION 1
SECTION II - PERFORMANCE 2
SECTION III - COST 3
SECTION IV - ORGANIZATION 4
SECTION V - CIRCUIT CHARACTERISTICS 9
SECTION VI - MECHANIZATION EQUATIONS 11
SECTION VII - ABBREVIATIONS 22
SECTION VIII - FLOATING POINT SUBROUTINES 26

LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure Number</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Fast Serial Digital Computer</td>
<td>5</td>
</tr>
<tr>
<td>2</td>
<td>Typical Timing Circuit Configuration</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>Signal Propagation Time</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>Notation for Logical Diagrams</td>
<td>30</td>
</tr>
<tr>
<td>5</td>
<td>A Register</td>
<td>31</td>
</tr>
<tr>
<td>6</td>
<td>A Gates I</td>
<td>32</td>
</tr>
<tr>
<td>7</td>
<td>A Gates II</td>
<td>33</td>
</tr>
<tr>
<td>8</td>
<td>A_T Gating</td>
<td>34</td>
</tr>
<tr>
<td>9</td>
<td>Adder</td>
<td>35</td>
</tr>
<tr>
<td>10</td>
<td>Miscellaneous Controls</td>
<td>36</td>
</tr>
<tr>
<td>11</td>
<td>Divide Control</td>
<td>37</td>
</tr>
<tr>
<td>12</td>
<td>Q Register</td>
<td>38</td>
</tr>
<tr>
<td>13</td>
<td>Q Gates I</td>
<td>39</td>
</tr>
<tr>
<td>Figure Number</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>---------------</td>
<td>--------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>14</td>
<td>Q Gates II</td>
<td>40</td>
</tr>
<tr>
<td>15</td>
<td>Q_</td>
<td>41</td>
</tr>
<tr>
<td>16</td>
<td>B Register</td>
<td>42</td>
</tr>
<tr>
<td>17</td>
<td>Shift Counter</td>
<td>43</td>
</tr>
<tr>
<td>18</td>
<td>Shift Counter Controls</td>
<td>44</td>
</tr>
<tr>
<td>19</td>
<td>Program Counter</td>
<td>45</td>
</tr>
<tr>
<td>20</td>
<td>Program Counter Gates</td>
<td>46</td>
</tr>
<tr>
<td>21</td>
<td>Index Register A</td>
<td>47</td>
</tr>
<tr>
<td>22</td>
<td>G Register</td>
<td>48</td>
</tr>
<tr>
<td>23</td>
<td>Index Selection</td>
<td>49</td>
</tr>
<tr>
<td>24</td>
<td>Index Adder (XAD)</td>
<td>50</td>
</tr>
<tr>
<td>25</td>
<td>Memory Address Storage Gates</td>
<td>51</td>
</tr>
<tr>
<td>26</td>
<td>Memory Address Storage Register -</td>
<td>52</td>
</tr>
<tr>
<td></td>
<td>Memory Address Register</td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>Memory Input-Output Register</td>
<td>53</td>
</tr>
<tr>
<td>28</td>
<td>Memory Input-Output and Memory Buffer Register</td>
<td>54</td>
</tr>
<tr>
<td></td>
<td>Stage 1 of 24</td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>MIO-MB Controls</td>
<td>55</td>
</tr>
<tr>
<td>30</td>
<td>Memory Input-Output Register Shift Control</td>
<td>56</td>
</tr>
<tr>
<td>31</td>
<td>Parity</td>
<td>57</td>
</tr>
<tr>
<td>32</td>
<td>Memory Controls</td>
<td>58</td>
</tr>
<tr>
<td>33</td>
<td>Program Interrupt Controls</td>
<td>59</td>
</tr>
<tr>
<td>34</td>
<td>Program Interrupt Register</td>
<td>60</td>
</tr>
<tr>
<td>35</td>
<td>Instruction Register</td>
<td>61</td>
</tr>
<tr>
<td>36</td>
<td>Decoder</td>
<td>62</td>
</tr>
<tr>
<td>Figure Number</td>
<td>Description</td>
<td>Page</td>
</tr>
<tr>
<td>---------------</td>
<td>------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>37</td>
<td>F Counter</td>
<td>63</td>
</tr>
<tr>
<td>38</td>
<td>F Functions</td>
<td>64</td>
</tr>
<tr>
<td>39</td>
<td>T Counter</td>
<td>65</td>
</tr>
<tr>
<td>40</td>
<td>Lambda Flip Flop</td>
<td>66</td>
</tr>
<tr>
<td>41</td>
<td>I/O Controls</td>
<td>67</td>
</tr>
<tr>
<td>42</td>
<td>I/O Controls</td>
<td>68</td>
</tr>
<tr>
<td>43</td>
<td>Word Buffer Register</td>
<td>69</td>
</tr>
<tr>
<td>44</td>
<td>Address Counter</td>
<td>70</td>
</tr>
<tr>
<td>45</td>
<td>Word Counter</td>
<td>71</td>
</tr>
<tr>
<td>46</td>
<td>Character Counter</td>
<td>72</td>
</tr>
<tr>
<td>47</td>
<td>Device Buffer</td>
<td>73</td>
</tr>
</tbody>
</table>
This report presents a design for a fast serial digital computer. This design was made to answer the questions:

What is currently feasible for this type of computer?

What characteristics might such a machine have?

The logic circuits used are the 100 mc circuits developed at MITRE. The memory is a parallel magnetic storage unit with 0.5-µs cycle time. Ten delay lines (developed at MITRE) are used for one-word registers.

In Sections II and III, the performance of the computer and the amount of hardware required are summarized. The remainder of the report is devoted to the detailed design.
SECTION II

PERFORMANCE

Assume a 100-mc clock rate and a 0.5-μs memory. Each word consists of 24 usable bits including sign. The time required for typical instructions is:

- Add: 1 μs
- Shift: 1 μs (either direction, any amount)
- Multiply: 6.75 μs
- Divide: 6.75 μs
- Normalize: 1.25 μs (transfers control if number is zero)

All other built-in instructions require 1 μs (load, store, etc.).

Floating point operations are not built-in, but can be done by subroutine. Such subroutines have been written with the following average times:

- Floating add: 24 μs
- Floating multiply: 15 μs
- Floating divide: 25 μs

Three index registers are provided. A program interrupt feature is included. A parity bit is generated for each word stored in memory. When a word is read from memory, its parity is checked.

There is a single channel which can be used for both input and output, but not both simultaneously. Once initiated, an I/O operation can transfer any given number of words between the I/O device and the memory without attention from the main program. When the I/O operation is complete, a program interrupt signal is generated.
Table I contains the number of circuit packages used for each type of logic circuit.

### Table I

#### Circuit Packages

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Figure No. in Appendix</th>
<th>AND Gates</th>
<th>OR Gates</th>
<th>FF</th>
<th>I &amp; A</th>
</tr>
</thead>
<tbody>
<tr>
<td>A register</td>
<td>5-8</td>
<td>65</td>
<td>4</td>
<td>10</td>
<td>1.5</td>
</tr>
<tr>
<td>Arithmetic</td>
<td>9-11</td>
<td>34</td>
<td>3</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>Q register</td>
<td>12-15</td>
<td>46</td>
<td>2</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>B register</td>
<td>16</td>
<td>12</td>
<td>2</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>Shift counter</td>
<td>17-18</td>
<td>41</td>
<td>3</td>
<td>5</td>
<td>1.5</td>
</tr>
<tr>
<td>PC</td>
<td>19-20</td>
<td>20</td>
<td>2</td>
<td>3</td>
<td>0.5</td>
</tr>
<tr>
<td>Index regs, G</td>
<td>21-24</td>
<td>77</td>
<td>3</td>
<td>16</td>
<td>2</td>
</tr>
<tr>
<td>MAS, MA</td>
<td>25-26</td>
<td>56</td>
<td>0</td>
<td>30</td>
<td>3</td>
</tr>
<tr>
<td>MIO, MB</td>
<td>27-32</td>
<td>199</td>
<td>5</td>
<td>56</td>
<td>15</td>
</tr>
<tr>
<td>PI</td>
<td>33-34</td>
<td>27</td>
<td>2</td>
<td>7</td>
<td>1</td>
</tr>
<tr>
<td>IR, decoder</td>
<td>35-36</td>
<td>56</td>
<td>2</td>
<td>12</td>
<td>5.5</td>
</tr>
<tr>
<td>Timing</td>
<td>37-40</td>
<td>45</td>
<td>1</td>
<td>15</td>
<td>26.5</td>
</tr>
<tr>
<td>I/O</td>
<td>41-47</td>
<td>97</td>
<td>4</td>
<td>27</td>
<td>3.5</td>
</tr>
<tr>
<td><strong>GATES</strong></td>
<td></td>
<td>775</td>
<td>33</td>
<td>195</td>
<td>62</td>
</tr>
<tr>
<td><strong>PKGS</strong></td>
<td></td>
<td>388</td>
<td>33</td>
<td>195</td>
<td>62</td>
</tr>
</tbody>
</table>

- Logic packages: 678
- Clock pulse packages: 88
- **TOTAL**: 766 packages

Other: 10 delay lines with drive amplifiers and phase inverters. 1 memory, 0.5-µs cycle, 25 bits, up to 32 K.

* I represents inverting amplifiers

A represents noninverting amplifiers
SECTION IV
ORGANIZATION

The block diagram of the fast serial digital computer is shown in Figure 1. The memory is connected to two registers of flip-flops, MB and MIO. The MIO is also a shift register, and it is here that the parallel to serial conversion takes place.

The A register is the main arithmetic register. The Q register is used in multiply and divide. Both are delay lines, as are the three index registers, XRA, XRB, and XRC.

The MAS and MA are flip-flop registers used for the memory address. The program counter (PC) is a delay line.

The I/O circuitry includes three delay lines, the work counter (WC), address counter (AC), and word buffer (WB). The device buffer (DB) is an eight-bit flip-flop register.

Word format is as follows:

**Data:**

- Bits 0–22: magnitude
- Bit 23: sign (zero for plus)
- Bit 24: guard bit (not usable).

Ones complement is used for negative numbers.

**Instructions:**

- Bits 0–14: address
- Bits 15–17: index tag
- Bits 18–23: operation code
- Bit 24: guard bit.
Figure 1. Fast Serial Digital Computer
In memory storage the guard bit is replaced by a parity bit.

One word time is 250 ns. Normally an instruction requires four word times (1 µs), i.e., two memory cycles. These four word times are designated T1 to T4. Within each word time there are 25 bit times, designated F₀ to F₂₄. The bit time (Tᵢ and Fⱼ) is abbreviated Ti, in the mechanization equations.

In preparation for starting a computer program, a master reset signal is provided. This signal:

1. Clears I/O control flip-flops;
2. Clears AC, WC, CC, and DB;
3. Clears λ and injects HLT into IR;
4. Sets the program interrupt controls to accept only the "input complete" interrupt, and clears PI, IC.

(It also destroys the contents of most registers by suspending the t pulses.) A console signal can then set the input status FF (IS) to initiate an input operation. This operation will fill memory, starting with location 0, until an end-of-file signal sets PI, IC. This causes the computer to execute the instruction in location 1.
<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LDX</td>
<td>Load index register</td>
</tr>
<tr>
<td>LDQ</td>
<td>Load Q register</td>
</tr>
<tr>
<td>CLA</td>
<td>Clear and add into A</td>
</tr>
<tr>
<td>CLS</td>
<td>Clear and subtract</td>
</tr>
<tr>
<td>CAM</td>
<td>Clear and add magnitude</td>
</tr>
<tr>
<td>SHL</td>
<td>Shift left</td>
</tr>
<tr>
<td>SHR</td>
<td>Shift right</td>
</tr>
<tr>
<td>CYR</td>
<td>Cycle right</td>
</tr>
<tr>
<td>NMT</td>
<td>Normalize and transfer</td>
</tr>
<tr>
<td>NEG</td>
<td>Complement A</td>
</tr>
<tr>
<td>LGS</td>
<td>Logical sum</td>
</tr>
<tr>
<td>LGM</td>
<td>Logical multiply</td>
</tr>
<tr>
<td>STA</td>
<td>Store A</td>
</tr>
<tr>
<td>STQ</td>
<td>Store Q</td>
</tr>
<tr>
<td>STX</td>
<td>Store index</td>
</tr>
<tr>
<td>STZ</td>
<td>Store zero</td>
</tr>
<tr>
<td>ADD</td>
<td>Add</td>
</tr>
<tr>
<td>SUBT</td>
<td>Subtract</td>
</tr>
<tr>
<td>ADM</td>
<td>Add magnitude</td>
</tr>
<tr>
<td>SBM</td>
<td>Subtract magnitude</td>
</tr>
<tr>
<td>MLY</td>
<td>Multiply</td>
</tr>
<tr>
<td>DIV</td>
<td>Divide</td>
</tr>
<tr>
<td>TRU</td>
<td>Transfer unconditionally</td>
</tr>
<tr>
<td>TRN</td>
<td>Transfer if negative</td>
</tr>
<tr>
<td>TRP</td>
<td>Transfer if positive</td>
</tr>
<tr>
<td>TRZ</td>
<td>Transfer if zero</td>
</tr>
<tr>
<td>TROV</td>
<td>Transfer if overflow</td>
</tr>
<tr>
<td>TRX</td>
<td>Transfer and decrement index</td>
</tr>
<tr>
<td>TSX</td>
<td>Transfer and set index</td>
</tr>
<tr>
<td>RPI</td>
<td>Read program interrupt requests</td>
</tr>
<tr>
<td>SPI</td>
<td>Set program interrupt controls</td>
</tr>
<tr>
<td>SIO</td>
<td>Start input/output</td>
</tr>
<tr>
<td>HLT</td>
<td>Halt</td>
</tr>
<tr>
<td>NOP</td>
<td>No operation</td>
</tr>
</tbody>
</table>
### Memory Timing

<table>
<thead>
<tr>
<th></th>
<th>Memory</th>
<th>PC/MAS</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Normal</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T3</td>
<td>Read data</td>
<td>MAS → PC if branch</td>
</tr>
<tr>
<td>T4</td>
<td>Write data</td>
<td>PC → MAS</td>
</tr>
<tr>
<td>T1</td>
<td>Read instruction</td>
<td>PC+1 → PC</td>
</tr>
<tr>
<td>T2</td>
<td>Write instruction</td>
<td>Index into MAS</td>
</tr>
<tr>
<td><strong>Program Interrupt</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T3</td>
<td>Read data</td>
<td></td>
</tr>
<tr>
<td>T4</td>
<td>Write data</td>
<td></td>
</tr>
<tr>
<td>T1</td>
<td>Clear 0</td>
<td></td>
</tr>
<tr>
<td>T2</td>
<td>Write PC</td>
<td></td>
</tr>
<tr>
<td>T1</td>
<td>Read instruction in 1</td>
<td></td>
</tr>
<tr>
<td>T2</td>
<td>Write instruction</td>
<td></td>
</tr>
<tr>
<td><strong>I/O Cycle</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T3</td>
<td>Read data</td>
<td></td>
</tr>
<tr>
<td>T4</td>
<td>Write data</td>
<td></td>
</tr>
<tr>
<td>T1</td>
<td>Read I/O word</td>
<td></td>
</tr>
<tr>
<td>T2</td>
<td>Write I/O word</td>
<td></td>
</tr>
<tr>
<td>T1</td>
<td>Read instruction</td>
<td></td>
</tr>
<tr>
<td>T2</td>
<td>Write instruction</td>
<td></td>
</tr>
</tbody>
</table>
SECTION V

CIRCUIT CHARACTERISTICS

The timing assumption is as follows:

For any signal subject to change, at least every fourth gate must be an AND gate with a clock pulse input. (An exception is the timing levels, for which special methods are used.) The typical configuration is shown in Figure 2.

![Figure 2. Typical Timing Circuit Configuration](image)

For example if

(1) the clock pulse has a rise time of 1 ns;

(2) the logic circuits have rise and fall times of 2 ns; and

(3) the logic circuits have propagation delays of 2 ns;

then we have the situation in Figure 3.
Figure 3. Signal Propagation Time

The level at B must fall before the pulse at B rises. Thus there is only 0.5 ns available in the above situation for point-to-point propagation, assuming a 100 mc clock rate.

But the assumptions are only a crude approximation to the actual characteristics of the logic circuits. Only a detailed study of the timing situation will determine the maximum permissible clock rate.

The maximum fan-out is six. There is the possibility that this is overly conservative, and that in fact a fan-out of eight (for levels) could be used. The fan-out restriction implies that certain circuits must be duplicated. Duplication is often preferable to using amplifiers, which increase the number of stages of delay.
SECTION VI
MECHANIZATION EQUATIONS

The notation used is: When the conditions to the left of the colon are satisfied, then the action specified on the right side is taken. The index of abbreviations is in Section VII.

Memory

1. \((T_{123}^{\prime})(SI')\) : [0 => MIO]
   \((T_{123}^{\prime})(SI)\) : [0 => MB]
   \((T_{124}^{\prime})(SI')\) : [MB => MIO]
   \((T_{124}^{\prime})(SI)\) : [MIO => MB]
   \((T_{124}^{\prime})\) : [1 \rightleftharpoons WR] [0 \rightleftharpoons SI] Rewrite instruction

2. \((T_{223}^{\prime})\) : [0 => MA]
   \((T_{24}^{\prime})\) : [MAS => MA] [1 \rightleftharpoons RD] [0 => MB] Read data

3. \((T_{323}^{\prime})(SI')\) : [0 => MIO]
   \((T_{324}^{\prime})(SI')\) : [MB => MIO]
   \((T_{324}^{\prime})\) : [1 \rightleftharpoons WR] [0 \rightleftharpoons SI] Rewrite data

4. \((T_{423}^{\prime})(\lambda')\) : [0 => MA]
   \((T_{424}^{\prime})(\lambda')\) : [MAS => MA] [1 \rightleftharpoons RD] [0 => MB] Read Instruction

Instruction Access

1. \((T_{40-14}^{\prime})(IIC')\) : [PC \rightleftharpoons MAS]
2. \((T_{424}^{\prime})(IIC')(\lambda')\) : [1 \rightleftharpoons IPC]
3. \((T^2_{0-14})(IIC')\) : [Index adder → MAS]

4. \((T^2_{23})(IIC')\) : [MB\(_{18-23}\) => IR] Double-line

Note: IIC = IOF + PI + HLT.

**Program Interrupt**

1. \((T^3_{24})(IOF')(IPI')\) \(\{(PI, IC) + (\Sigma PI)(PIC)\}\) : [1 → IPI] [1 → PIC]

2. \((PI)(T^4_{0-14})\) : [0 → MAS]

3. \((PI)(T^4_{24})\) : [1 → SI]

4. \((PI)(TI)\) : [PC → MIO]

5. \((PI)(T^2_{22})\) : [1 → λ]

\((PI)(T^2_{23})\) : [1 → MAS\(_0\)] [NOP => IR] [1 → T] [0 → λ]

\((PI)(T^2_{24})\) : [0 → PIC]

Note: PI = (IPI)(PIC)

Note: The instruction in location 1 should be a TRU, since the above does not alter PC. The instruction whose address is stored in location 0 has not yet been executed.

**Input**

1. \((PI')(IDR)(T^2_{23})\) : [1 → ICF]

2. \((ICF)(T^3)(CCF)\) : [DB → WB] [0 → DB]

3. \((ICF)(T^3_{23})\) : [CC+1 → CC] [0 → IDR] [0 → ICF]

\((ICF) (T^3_{23})(CC=2)\) : [1 → IF] [1 → IAC, DWC]
Output

1. \((P1')(ODR)(T2_{23}) : [1 \rightarrow OCF]\]
2. \((OCF)(T3)(CCF) : [WB \rightarrow DB]\]
3. \((OCF)(T3_{23}) : [CC+1 \rightarrow CC][0 \rightarrow ODR][0 \rightarrow OCF]\)
   \((OCF)(T3_{23})(CC=2)(WC\neq0) : [1 \rightarrow OF]\)
   \((OCF)(T3_{23})(CC=2)(WC=0) : [1 \rightarrow PI, OC][0 \rightarrow OS]\)

Note: CC counts modulo 3.

CCF = \((CC=0)(F_{0-7}) + (CC=1)(F_{8-15}) + (CC=2)(F_{16-23})\)

Note: Device sets IDR or ODR.

Clearing IDR or ODR allows device to proceed to fill or read DB.

Note: DB shift signal = \((T3)(CCF)(ICF + OCF)\).

Input/Output

1. \((IOF)(T4_{0-14}) : [AC \rightarrow MAS]\]
2. \((IF)(T4_{24}) : [1 \rightarrow SI]\]
3. \((IF)(T1) : [WB \rightarrow MIO]\]
4. \((IOF)(T2_{0-14}) : [PC \rightarrow MAS]\]
5. \((OF)(T2) : [MIO \rightarrow WB]\]
6. \((IOF)(T2_{22}) : [1 \rightarrow \lambda]\)
   \((IOF)(T2_{23})(HLT') : [NOP \rightarrow IR]\)
7. \((IOF)(T2_{23}) : [1 \rightarrow T][0 \rightarrow \lambda]\)
   \((IOF)(T2_{24}) : [0 \rightarrow IOF]\)
(IF)(T2_{24})(WC=0) : [ 0 → IS] [ 1 → PI, IC]
(OF)(T2_{24}) : [ 1 → IAC, DWC]
(IFO)(T2_{24})(HLT') : [ 1 → IPC]

Miscellaneous

1. (T4_{24})\(\lambda\)': [ 0 → IMS]
2. (T4) : [MIO → B] unless specifically overruled
3. (T1_{24}) : [ 0 → SN]
4. (F23) : [ 0 → SUB]
5. (T4)(\lambda) : [ B → B]

Instructions

LDX
1. (T1) : [B → XRG]

LDQ
1. (T1) : [B → Q]

CLA
1. (T1) : [B → A]

CLS
1. (T4) : [ MIO' → B]
2. (T1) : [ B → A]

CAM
1. (MB_{23})(T4) : [ MIO' → B]
2. (T1) : [ B → A]
SHL
1. \((T_{24}^2) : [1 \rightarrow SI]\)
2. \((T3) : [A \rightarrow MIO][0 \rightarrow A]\)
3. \((T3_{23}) : [1 \rightarrow IMS]\)
4. \((T4) : [SC-1 \rightarrow SC]\)
5. \((T4)(SC=0) : [0 \rightarrow IMS]\)
6. \((T4)(IMS') : [MIO \rightarrow A]\)

SHR
1, 2, 4 as in SHL.
8. \((T4_{0-23})(SC=0) : [1 \rightarrow IMS]\)
9. \((T1) : [MIO \rightarrow A]\)

Note: Copies of sign enter from the left.

CYR
1, 2, 4, 8, 9 as above.
7. \((T4) : [MIO \rightarrow MIO]\)

Note: The three above instructions shift one place more than
the number in the address field.

NMT Normalize and Transfer
1. \((T_{24}^2) : [-0 => SC][1 \rightarrow SI]\)
2. \((T3_{0-14})(A_{-Z}) : [MAS \rightarrow PC]\)
3. \((T3) : [A \rightarrow MIO]\)
4. \( (T_{30-22}^3)(A \neq A_{SN}) : [-0 \Rightarrow SC] \)
\( (T_{0-22}^3)(A = A_{SN}) : [SC-1 \Rightarrow SC] \)
5. \( (T_{323}^3) : [1 \rightarrow IMS] \)
6. \( (T_{40-4}^4)(\lambda') : [SC \leftarrow A][SC' \leftarrow SC] \)
\( (T_{45-24}^4)(\lambda') : [1 \rightarrow A] \)
7. \( (T_{422}^4)(\lambda') : [1 \rightarrow \lambda] \)
8. \( (T_{424}^4)(SC=0) : [0 \rightarrow IMS] \)
\( (T_{4}^4)(\lambda)(SC=1) : [0 \rightarrow IMS] \)
9. \( (T_{4}^4)(\lambda) : [SC-1 \Rightarrow SC] \)
10. \( (T_{4}^4)(IMS) : [MIO_{23} \rightarrow Q] \)
\( (T_{4}^4)(IMS') : [MIO \rightarrow Q] \)
11. \( (T_{422}^4)(\lambda) : [0 \rightarrow \lambda] \)

Note: A contains the negative of the number of shifts. The normalized number (if non-zero) is in Q.

**NEG**

1. \( (T_{3}^3) : [A' \rightarrow A] \)

**LGS** Logical Sum

1. \( (T_{1}^1) : [A \lor B \rightarrow A] \) (inclusive or)
LGM  Logical Multiply

1.  (T1)  :  [A \cdot B \rightarrow A]

STA  Store A

1.  (T_{24})  :  [1 \rightarrow SI]
2.  (T3)  :  [A \rightarrow MIO]
3.  (T_{23})  :  [0 \Rightarrow MB]
4.  (T_{24})  :  [MIO \Rightarrow MB]

STQ  Store Q
1, 3, 4 as above.

2.  (T3)  :  [Q \rightarrow MIO]

STZ  Store Zero
1, 3, 4 as above
2.  (T3)  :  [0 \rightarrow MIO]

STX  Store Index
1, 3, 4 as in STA.
2.  (T3)  :  [XRG \rightarrow MIO]

ADD

1.  (T_{24})  :  [\text{A}^{_{SN}} \rightarrow SN]
2.  (T1 + T2)  :  [A \oplus B \rightarrow A]  \text{ Arithmetic sum}
3.  (T_{23})  :  [\text{SN} = \text{B}^{_{SN}} \neq A \oplus B]  :  [1 \rightarrow OV]
SUBT

0. \((T4) : [MIO' \rightarrow B]\)

1, 2, 3 as above.

ADM

0. \((MB_{23})(T4) : [MIO' \rightarrow B]\)

1, 2, 3 as above.

SBM

0. \((MB_{23}')\)(T4) : [MIO' \rightarrow B]

1, 2, 3 as above.

MLY

1. \((T_{23}^3)(A_{SN} \neq MB_{23}) : [1 \rightarrow SN]\)

\((T_{23}^3) : [-0 \Rightarrow SC]\)

2. \((T4)(\lambda') : [0 \rightarrow A]\)

\((TR)(\lambda') (A_{SN}') : [A \rightarrow Q]\)

\((T4)(\lambda')(A_{SN}) : [A' \rightarrow Q]\)

\((T4)(\lambda')(MB_{23}) : [MIO' \rightarrow B]\)

3. \((T_{22}^4)(A_{Z'})(\lambda') : [1 \rightarrow \lambda]\)

\((T_{23}^4) : [SC-1 \Rightarrow SC]\)

\((T_{22}^4)(SC = 23) : [0 \rightarrow \lambda]\)

4. \((\lambda')(F_{0-21}) : [Q_T \rightarrow Q]\)

\((\lambda')(F_0')(Q_0) : [A \oplus B \rightarrow Q_+]\)
\( (\lambda)(F_0')(Q_0') : [A \rightarrow Q_+] \)
\( (\lambda)(F_0')(Q_0) : [A \oplus B \rightarrow A_T] \)
\( (\lambda)(F_0')(Q_0')' : [A \rightarrow A_T] \)
\( (\lambda)(F_{22}) : [Q_+ \rightarrow Q] \)
\( (\lambda)(F_{23-24}) : [0 \rightarrow Q] \)
\( (\lambda)(F_{24}) : [Q_T \rightarrow Q_0] \) \text{(normal)}

5. \( (T1)(SN) : [A' \rightarrow A] \) \( [Q' \rightarrow Q] \)

Note: During \((MLY)(\lambda), the normal Q_T \rightarrow Q_0 \) is suppressed.

\textbf{DIV}

1. \( (T3)(A_{SN'}) : [A' \rightarrow A] \) \( [Q' \rightarrow Q] \)
\( (T3_{23}) (A_{SN} \neq MB_{23}) : [1 \rightarrow SN] \)
\( (T3_{23}) : [-0 \Rightarrow SC]\) \( [0 \rightarrow Q_+] \)

2. \( (\lambda')(MB_{23})(T4) : [MIO' \rightarrow B] \)

3. \( (\lambda')(T4_{22}) : [1 \rightarrow \lambda] \)
\( (T4_{23}) : [SC-1 \Rightarrow SC] \)
\( (T4_{23}) (SC = -23) : [0 \rightarrow \lambda] \)
\( (\lambda)(F_{24})(A'_+) : [1 \rightarrow OV] \) \hspace{1cm} \text{Abort if}
\( (\lambda)(F_{22})(OV) : [0 \rightarrow \lambda] \) \hspace{1cm} \text{overflow}

4. \( (\lambda)(SUB) : [A \oplus B \rightarrow A_+] \)
\( (T4)(SUB') : [A \rightarrow A_+] \)

5. \( (T4_{0})(SC \neq -23) : [Q_H \rightarrow A] \)
\( (T4_{1-24})(SC \neq -23) : [A_+ \rightarrow A] \)

19
6. \((T4) : [Q \rightarrow Q_+ ]\)
\((T4_0) : [SUB \rightarrow Q]\)
\((T4_1-24) : [Q_+ \rightarrow Q]\)

7. \((T4_{24})(B \leq |A| ) : [1 \rightarrow SUB]\)

8. \((SC = -23)(SUB) : [A \oplus B \rightarrow A]\)
\((SC = -23)(SUB')(\lambda) : [A \rightarrow A]\)

9. \((T1)(SN') : [Q' \rightarrow Q]\)

Note: Remainder in A is negative.

TRU

1. \((T3_{0-14}) : [MAS \rightarrow PC]\)

TRN

1. \((T3_{0-14})(A_{SN}) : [MAS \rightarrowPC]\)

TRP

1. \((T3_{0-14})(A_{SN'}) : [MAS \rightarrow PC]\)

TRZ

1. \((T3_{0-14})(A_Z) : [MAS \rightarrow PC]\)

TROV

1. \((T3_{0-14})(0V) : [MAS \rightarrow PC]\)

2. \((T3_{24}) : [0 \rightarrow 0V]\)

TRX

1. \((T3_{0-14})(XRGZ') : [MAS \rightarrow PC]\)

2. \((T3_{24})(XRGZ') : [1 \rightarrow DXRG]\)
TSX
1. \((T3_{0-14})\) : \([\text{MAS} \to \text{PC}] [\text{PC} \to \text{XRG}]\)

RPI Read Program Interrupt Requests
1. \((T3_{0-4})\) : \([\text{PIR} \to A][0 \to \text{PIR}]\)
2. \((T3_{5-24})\) : \([0 \to A]\)

Note: No FF in PIR should be set during \(F_0 - F_4\).

Last FF in PIR should not be set during \(F_0 - F_{22}\).

SPI Set Program Interrupt Controls
1. \((T3_{24})\) : \([\text{MA}_0 \to \text{PIC}] [\text{MA}_1 \to \text{IPI}]\)

Note: The effect of these controls is:

\((\text{IPI'}) (\text{PIC})\) : Any interrupt will be recognized.

\((\text{IPI'}) (\text{PIC'})\) : Only PI, IC will be recognized.

\((\text{IPI}) (\text{PIC'})\) : No interrupts will be recognized.

Note: After clearing IPI, at least one instruction is executed before the next interrupt is recognized.

SIO Start input/output
1. \((T3)\) : \([\text{MAS} \to \text{AC}] [\text{A} \to \text{WC}]\)
2. \((A_Z') (A_{SN}) (T3_{23})\) : \([1 \to \text{OS}] [1 \to \text{OF}]\)

\((A_{SN}') (T3_{23})\) : \([1 \to \text{IS}]\)

HLT Halt
1. \((T2_{23})\) (Start Button) : \([\text{NOP} \to \text{IR}]\)

Note: A program interrupt will also start machine.
SECTION VII

ABBREVIATIONS

The number in parentheses represents the most relevant illustration.

A, A_{SN}, A_{T}, etc. (5).
AC. Address counter.
ADD. An instruction.
Add class. ADD + SUBT + ADM + SBM (36).
ADM. Add magnitude (instruction).
B, B_{SN}, B_{24}. (16).
CAM. Clear and add magnitude (instruction).
CC. Character count (46).
CCF. (46).
CLA. Clear and add (instruction).
CLS. Clear and subtract (instruction).
COA. Complementary output amplifier (4).
CYR. Cycle right (instruction).
DA. Drive amplifier (4).
DB. Device buffer (47).
DIV. Divide (instruction).
DL. Delay line (4).
DWC. Decrease word count (45).
DXR(A). Decrease index register (A) (21).
EOF. End of file (34).
F_{i}. Timing function number i (38).
FF. Flip-flop (4).
G. (22).
HLT. Halt (instruction).
I. Inverter (4).
IAC. Increase address counter (44).
ICF. Input control FF (41).
IDR. Input data ready (41).
IF. Input flag (41).
IIC. Inhibit instruction control (35).
IMS. Inhibit MIO shift (30).
IOF. Input/output flag (41).
IP. Input parity (31).
IPC. Increase program counter (19).
IPI. Inhibit program interrupt (33).
IR. Instruction register (35).
IS. Input status (42).
LDX. Load index (instruction).
LDQ. Load Q register (instruction).
LGM. Logical multiply (instruction).
LGS. Logical sum (instruction).
MA. Memory address register (26).
MAS. Memory address storage (26).
MB. Memory buffer register (28).
MIO. Memory input-output register (28).
MLY. Multiply (instruction).
MRE. Master reset signal.
NEG. Negate (instruction).
NMT. Normalize and transfer (instruction).
NOP. No operation (instruction).
OCF. Output control FF (41).
ODR. Output data ready (41).
OF. Output flag (42).
OP. Output parity (31).
OS. Output status (42).
OV. Overflow (10).
PC. Program counter (19).
PI. Program interrupt level (33).
PIC. Program interrupt control (33).
PIR. Program interrupt register (34).
Q, Q^H, etc. (12).
RD. Read signal (29).
RPI. Read program interrupt requests (instruction).
SBM. Subtract magnitude (instruction).
SC. Shift counter (17).
Shift. SHL + SHR + CYR (36).
SHL. Shift left (instruction).
SHR. Shift right (instruction).
SI. Strobe inhibit (32).
SIO. Start input/output operation (instruction).
SN. Sign (10).
SPI. Set program interrupt controls (instruction).
STA. Store A (instruction).
Store. STA + STQ + STX + STZ (36).
STQ. Store Q (instruction).
STX. Store index (instruction).
STZ. Store zero (instruction).
SUB. Subtraction control (33).
SUBT. Subtract (instruction).
t. Clock pulse.
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ti.</td>
<td>Timing level (39).</td>
</tr>
<tr>
<td>TP.</td>
<td>Test parity (31).</td>
</tr>
<tr>
<td>TRN.</td>
<td>Transfer if A is negative (instruction).</td>
</tr>
<tr>
<td>TROV.</td>
<td>Transfer on overflow (instruction).</td>
</tr>
<tr>
<td>TRP.</td>
<td>Transfer if A is positive (instruction).</td>
</tr>
<tr>
<td>TRU.</td>
<td>Transfer unconditionally (instruction).</td>
</tr>
<tr>
<td>TRX.</td>
<td>Transfer and decrement index (instruction).</td>
</tr>
<tr>
<td>TRZ.</td>
<td>Transfer if A is zero (instruction).</td>
</tr>
<tr>
<td>TSX.</td>
<td>Transfer and set index (instruction).</td>
</tr>
<tr>
<td>WB.</td>
<td>Word buffer (43).</td>
</tr>
<tr>
<td>WC.</td>
<td>Word counter (45).</td>
</tr>
<tr>
<td>WCZ.</td>
<td>Word count is zero (45).</td>
</tr>
<tr>
<td>WR.</td>
<td>Write signal (32).</td>
</tr>
<tr>
<td>X.</td>
<td>Index signal (23).</td>
</tr>
<tr>
<td>XAD.</td>
<td>Index added to address (24).</td>
</tr>
<tr>
<td>XR(A)</td>
<td>Index register (A) (21).</td>
</tr>
<tr>
<td>XRG.</td>
<td>The index register specified by G.</td>
</tr>
<tr>
<td>α</td>
<td>Amplifier (2).</td>
</tr>
<tr>
<td>λ</td>
<td>(40).</td>
</tr>
<tr>
<td>φI</td>
<td>Phase inverter (4).</td>
</tr>
<tr>
<td>ΣPI</td>
<td>(34).</td>
</tr>
</tbody>
</table>
SECTION VIII

FLOATING POINT SUBROUTINES

We give programs for multiply, divide, and add.

FLOATING MULTIPLY SUBROUTINE

\[ 2^{e_3} m_3 = \left( 2^{e_1} m_1 \right) \left( 2^{e_2} m_2 \right) \]

CLA \quad m_1 \quad \text{Average time 15 \( \mu \text{s} \)}
MLY \quad m_2
NMT \quad Z
STQ \quad m_3
ADD \quad e_1
ADD \quad e_2
STA \quad e_3

\text{TROV} \quad * +2
TRU \quad \text{exit}
TRN \quad \text{OV exit}

Z \quad STZ \quad m_3 \quad (\text{Zero or}
CLA \quad \text{min. exp.} \quad \text{underflow})
STA \quad e_3
TRU \quad \text{exit}

FLOATING DIVIDE SUBROUTINE

\[ 2^{e_3} m_3 = \left( 2^{e_1} m_1 \right) / \left( 2^{e_2} m_2 \right) \]

CLA \quad e_1 \quad \text{Average time 25 \( \mu \text{s} \)}
SUBT \quad e_2
FLOATING ADD SUBROUTINE

\[
e^3 m_3 = 2^1 m_1 + 2^2 m_2.
\]

Average time 24 \(\mu\)s.

\[
\begin{align*}
\text{CLA} & \quad e_1 \\
\text{STA} & \quad e_3 \\
\text{XCH} & \quad \text{CLA} e_2 \\
\text{STA} & \quad e_3
\end{align*}
\]
<table>
<thead>
<tr>
<th>Operation</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBT</td>
<td>m_1</td>
</tr>
<tr>
<td>TROV</td>
<td>NA</td>
</tr>
<tr>
<td>TRZ</td>
<td>No shift</td>
</tr>
<tr>
<td>TRN</td>
<td>XCH</td>
</tr>
<tr>
<td>SUBT</td>
<td>&quot;22&quot;</td>
</tr>
<tr>
<td>TRP</td>
<td>No add</td>
</tr>
<tr>
<td>ADD</td>
<td>SHR 21</td>
</tr>
<tr>
<td>STA</td>
<td>* +2</td>
</tr>
<tr>
<td>CLA</td>
<td>m_2</td>
</tr>
<tr>
<td>___</td>
<td></td>
</tr>
<tr>
<td>AD</td>
<td>ADD</td>
</tr>
<tr>
<td>_______</td>
<td></td>
</tr>
<tr>
<td>β</td>
<td>TROV</td>
</tr>
<tr>
<td>NMT</td>
<td>Zero</td>
</tr>
<tr>
<td>STQ</td>
<td>m_3</td>
</tr>
<tr>
<td>ADD</td>
<td>e_3</td>
</tr>
<tr>
<td>TROV</td>
<td>Zero</td>
</tr>
<tr>
<td>STA</td>
<td>e_3</td>
</tr>
<tr>
<td>TRU</td>
<td>exit</td>
</tr>
<tr>
<td>α</td>
<td>TRN</td>
</tr>
<tr>
<td>CLA</td>
<td>m_2</td>
</tr>
<tr>
<td>STA</td>
<td>m_3</td>
</tr>
<tr>
<td>TRU</td>
<td>exit</td>
</tr>
<tr>
<td>NA</td>
<td>CLA</td>
</tr>
<tr>
<td>STA</td>
<td>m_3</td>
</tr>
<tr>
<td>TRU</td>
<td>exit</td>
</tr>
<tr>
<td>No Shift</td>
<td>CLA</td>
</tr>
<tr>
<td>TRU</td>
<td>AD</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operation</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUBT</td>
<td>e_1</td>
</tr>
<tr>
<td>SUBT</td>
<td>&quot;22&quot;</td>
</tr>
<tr>
<td>TRP</td>
<td>NA</td>
</tr>
<tr>
<td>ADD</td>
<td>SHR 21</td>
</tr>
<tr>
<td>STA</td>
<td>* +2</td>
</tr>
<tr>
<td>CLA</td>
<td>m_2</td>
</tr>
<tr>
<td>___</td>
<td></td>
</tr>
<tr>
<td>ADD</td>
<td>m_2</td>
</tr>
<tr>
<td>TRU</td>
<td>β</td>
</tr>
<tr>
<td>___</td>
<td></td>
</tr>
<tr>
<td>AD</td>
<td>ADD</td>
</tr>
<tr>
<td>_______</td>
<td></td>
</tr>
<tr>
<td>β</td>
<td>TROV</td>
</tr>
<tr>
<td>NMT</td>
<td>Zero</td>
</tr>
<tr>
<td>STQ</td>
<td>m_3</td>
</tr>
<tr>
<td>ADD</td>
<td>e_3</td>
</tr>
<tr>
<td>TROV</td>
<td>Zero</td>
</tr>
<tr>
<td>STA</td>
<td>e_3</td>
</tr>
<tr>
<td>TRU</td>
<td>exit</td>
</tr>
<tr>
<td>α</td>
<td>TRN</td>
</tr>
<tr>
<td>CLA</td>
<td>m_2</td>
</tr>
<tr>
<td>STA</td>
<td>m_3</td>
</tr>
<tr>
<td>TRU</td>
<td>exit</td>
</tr>
<tr>
<td>NA</td>
<td>CLA</td>
</tr>
<tr>
<td>STA</td>
<td>m_3</td>
</tr>
<tr>
<td>TRU</td>
<td>exit</td>
</tr>
<tr>
<td>No Shift</td>
<td>CLA</td>
</tr>
<tr>
<td>TRU</td>
<td>AD</td>
</tr>
<tr>
<td>Zero</td>
<td>CLA</td>
</tr>
<tr>
<td>STA</td>
<td>e_3</td>
</tr>
<tr>
<td>STZ</td>
<td>m_3</td>
</tr>
<tr>
<td>TRU</td>
<td>exit</td>
</tr>
</tbody>
</table>
APPENDIX

ILLUSTRATIONS
Figure 4. Notation for Logical Diagrams

AND GATE

OR GATE

BUFFERED OR

FLIP-FLOP

INVERTER

AMPLIFIER

COMPLEMENTARY OUTPUT AMPLIFIER

DELAY LINE WITH DRIVE AMPLIFIER AND PHASE INVERTER

DELAY, 1.5 NS UNLESS SPECIFIED

TWO UNITS DRIVEN IN PARALLEL
Figure 5. A Register
Figure 6. A Gates I
Figure 7. A Gates II
Figure 8. $A_T$ Gating
A ⊕ B is arithmetic sum
AvB is logical sum

Figure 9. Adder
Figure 10. Miscellaneous Controls
Figure 11. Divide Control
Figure 12. Q Register
Figure 13. Q Gates 1
Figure 14. Q Gates II
Figure 15. $Q_+$
Figure 16. B Register
Figure 17. Shift Counter
Figure 18. Shift Counter Controls
Figure 19. Program Counter
Figure 20. Program Counter Gates
Figure 21. Index Register A
Figure 22. G Register
Figure 23. Index Selection
Figure 24. Index Adder (XAD)
Figure 25. Memory Address Storage Gates
Figure 26. Memory Address Storage Register – Memory Address Register
Figure 27. Memory Input-Output Register Input Gates
TRIPLE FF ON MIO₀

Figure 28. Memory Input-Output and Memory Buffer Register Stage
(1 of 24)
Figure 29. MIO-MB Controls
Figure 30. Memory Input-Output Register Shift Control
Figure 31. Parity
Figure 32. Memory Controls
Figure 33. Program Interrupt Controls
Figure 34. Program Interrupt Register
Figure 35. Instruction Register
33 INSTRUCTIONS ARE DECODED

Figure 36. Decoder
Figure 38. F Functions
DELAY TRIMMED SO THAT T LEVEL RISES 12.5 NS AFTER \( F_{23}(t) \) PULSE

Figure 39. T Counter
Figure 40. Lambda Flip Flop
Figure 41. I/O Controls
Figure 42. I/O Controls
Figure 43. Word Buffer Register
Figure 44. Address Counter
Figure 45. Word Counter
**Figure 47.** Character Counter

CCF = (CC = 0)(F₀-7) + (CC = 1)(F₈-15) + (CC = 2)(F₁₆-23)

Figures 46 and 47 illustrate the character counter and related components.
Figure 47. Device Buffer
The MITRE Corporation
Bedford, Mass.

Logical Design for Fast Serial Computer

N/A

Enderton, H. B.

July 1965

AF19(628)-2390
508

ESD-TR-65-81

W-07189

Qualified requestors may obtain from DDC.
DDC release to OTS authorized.

Directorate of Computers.
Electronic Systems Division
L. G. Hanscom Field, Bedford, Mass.

The detailed logical design is given for a serial digital computer using a 0.5-microsecond magnetic memory, 100-mc logical circuits, and one-word delay lines. The computer performs most instructions in 1 microsecond. A list is given of the amount of hardware used.
### Instructions

1. ORIGINATING ACTIVITY: Enter the name and address of the contractor, subcontractor, grantee, Department of Defense activity or other organization (corporate author) issuing the report.

2a. REPORT SECURITY CLASSIFICATION: Enter the overall security classification of the report. Indicate whether "Restricted Data" is included. Marking is to be in accordance with appropriate security regulations.

2b. GROUP: Automatic downgrading is specified in DoD Directive 5200.10 and Armed Forces Industrial Manual. Enter the group number. Also, when applicable, show that optional markings have been used for Group 3 and Group 4 as authorized.

3. REPORT TITLE: Enter the complete report title in all capital letters. Titles in all cases should be unclassified. If a meaningful title cannot be selected without classification, show title classification in all capitals in parenthesis immediately following the title.

4. DESCRIPTIVE NOTES: If appropriate, enter the type of report, e.g., interim, progress, summary, annual, or final. Give the inclusive dates when a specific reporting period is covered.

5. AUTHOR(S): Enter the name(s) of author(s) as shown on or in the report. Enter last name, first name, middle initial. If military, show rank and branch of service. The name of the principal author is an absolute minimum requirement.

6. REPORT DATE: Enter the date of the report as day, month, year; or month, year. If more than one date appears on the report, use date of publication.

7a. TOTAL NUMBER OF PAGES: The total page count should follow normal pagination procedures, i.e., enter the number of pages containing information.

7b. NUMBER OF REFERENCES: Enter the total number of references cited in the report.

8a. CONTRACT OR GRANT NUMBER: If appropriate, enter the applicable number of the contract or grant under which the report was written.

8b, 8c, & 8d. PROJECT NUMBER: Enter the appropriate military department identification, such as project number, subproject number, system numbers, task number, etc.

9a. ORIGINATOR’S REPORT NUMBER(S): Enter the official report number by which the document will be identified and controlled by the originating activity. This number must be unique to this report.

9b. OTHER REPORT NUMBER(S): If the report has been assigned any other report numbers (either by the originator or by the sponsor), also enter this number(s).

10. AVAILABILITY/LIMITATION NOTICES: Enter any limitations on further dissemination of the report, other than those imposed by security classification, using standard statements such as:

   (1) "Qualified requesters may obtain copies of this report from DDC."

   (2) "Foreign announcement and dissemination of this report by DDC is not authorized."

   (3) "U. S. Government agencies may obtain copies of this report directly from DDC. Other qualified DDC users shall request through "

   (4) "U. S. military agencies may obtain copies of this report directly from DDC. Other qualified users shall request through "

   (5) "All distribution of this report is controlled. Qualified DDC users shall request through "

If the report has been furnished to the Office of Technical Services, Department of Commerce, for sale to the public, indicate this fact and enter the price, if known.

11. SUPPLEMENTARY NOTES: Use for additional explanatory notes.

12. SPONSORING MILITARY ACTIVITY: Enter the name of the departmental project office or laboratory sponsoring (paying for) the research and development. Include address.

13. ABSTRACT: Enter an abstract giving a brief and factual summary of the document indicative of the report, even though it may also appear elsewhere in the body of the technical report. If additional space is required, a continuation sheet shall be attached.

   It is highly desirable that the abstract of classified reports be unclassified. Each paragraph of the abstract shall end with an indication of the military security classification of the information in the paragraph, represented as (TS), (S), (C), or (U).

   There is no limitation on the length of the abstract. However, the suggested length is from 150 to 225 words.

14. KEY WORDS: Key words are technically meaningful terms or short phrases that characterize a report and may be used as index entries for cataloging the report. Key words must be selected so that no security classification is required. Identifiers, such as equipment model designation, trade name, military project code name, geographic location, may be used as key words but will be followed by an indication of technical context. The assignment of links, rules, and weights is optional.

---

Digital Computers
Design
Logic