目 錄 CONTENTS

一、海明威主要作品主題之研究………………1 ～ 37 ……熊 暉 蕙
   STUDY ON THE THEMES
   OF ERNEST HEMINGWAY’S
   MAJOR WORKS
   Hsiung Hsiao-Huey

二、經驗貝氏最佳性的研討……………………39 ～ 57 ……朱 其 福
   THE EMPIRICAL BAYES
   OPTIMALITY
   Chyi-Fiuu Ju

三、利用V. YU. LEBEDEV近似法解線……59 ～ 64 ……鄭 彰 英
   型規劃問題之數值解
   AN APPROXIMATE ALGO-
   RITHM FOR THE NUMERICAL
   SOLUTION OF LINEAR
   PROGRAMMING PROBLEM
   BASED UPON V. YU. LEBE-
   DEV METHOD
   Cheng Chung-Ying

四、人體上肢型態與握力的相關研究…………65 ～ 101 ……鄭 文 義
   A STUDY OF THE CORRE-
   LATION OF THE TYPES OF
   THE UPPER EXTRIMITIES
   OF HUMAN BODY WITH
   HAND GRIP STRENGTH
   Wen-I Cheng

五、跳繩訓練對增進心血管性耐力之效果研…103 ～ 119 ……黃 學 勇
   THE EFFECT OF ROPE
   SKIPPING TO THE IN-
   CREASE OF CARDIVASCULAR
   ENDURANCE
   Huang Hsueh-Yun

六、三層夾模內共擠壓之層狀熔流流動……121 ～ 135 ……許 俊 顯
   STRATIFIED MELT FLOW
   IN SANDWICH SHEETING
   DIE OF CO-EXTRUSION
   Jin-Sheng Sheu

七、識圖分析………………………………………137 ～ 160 ……王 大 靜
   ANALYSIS OF READING
   ENGINEERING DRAWING
   Ta-Chine Wang
DIRECT STEPPING MOTOR
ACCELERATION AND POSITION CONTROL BY

A STUDY OF AUTOMATIC ELEVATOR CONTROLLED BY ELECTRONIC LOGIC CIRCUITS

ANALYTICAL TREATMENT ON THE STORED CHARGE AND TRANSIT TIME FOR EPITAXIAL PLANAR TRANSISTORS OPERATING IN SATURATION
DIRECT STEPPING MOTOR ACCELERATION AND POSITION CONTROL BY M6800 MICRO PROCESSOR

Kuo-En Huang (黃國恩)

INTRODUCTION

In the first part of this article we are going to design a program for a digital signal processor to generate a pulse rate proportional to a command voltage with controlled acceleration and polarity. The pulse output should be suitable for operating a stepping motor through a translator/power supply unit.

In the second part, we are going to control the stepping motor rotating angle accurately through the step counter and input pulse ramp interval assignment at the beginning and ending of the rotation.

Finally, we try to design a hardware interface which can be used to connect several stepping motors to one microprocessor to perform a parallel operation for position control.

HARDWARE DESCRIPTION

A. DIGITAL PROCESSOR

The signal processor is based on the MOTOROLA MEK 6800 D2 Evaluation Kit II. At the present time this processor contains approximately four and a half (4½) K bytes of RAM and one (1) K of EPROM. The processor is equipped with MOTOROLA'S JBUG monitor on ROM. The MPU clock is set at 614.4 KHZ. The processor is connected to peripherals through inverting buffers on both input and output lines, programs can be stored on audio cassette. There are three important additions besides extra memory to the Evaluation Kit which increase the power of the signal processor. Figure 1 shows a block diagram of the system.
B. ADDITIONS TO THE EVALUATION KIT

1. A/D Converter

The A/D converter is the BURR-BROWN MP 21 ANALOG INPUT MICROPERIPHERAL. The MP 21 is set up to accept an input of from \(-5\) to \(+5\) volts. The unit will output a two's complement number. The \(-5\) will be converted to a 80 hexadecimal, the \(+5\), to a 7F. The MP 21 requires at least 40 usec to produce a digital signal from an analog input. The MP 21 is initialized by loading zeroes into its internal register. There are 8 input channels.

2. D/A CONVERTER

The D/A converter is the BURR-BROWN MP 11 ANALOG OUTPUT MICROPERIPHERAL the MP11 converts a complementary binary number to an analog output voltage. For example, if a 00 hexadecimal number is stored in the MP 11 internal register, the MP 11 will output negative ten \((-10\)\) volts. Similarly, a FF hexadecimal number will produce a positive ten \((+10\)\) volts. The MP 11 requires 25 \(\mu\)sec to produce a stable output. It is initialized by loading its register with ones. There are 4 output channels.

C. USE OF PROCESSOR

Programs are entered in memory through a hexadecimal keyboard.

The keyboard is controlled by the JBUG monitor.

The peripheral devices are memory-mapped. TABLE 1 shows the addresses for the various peripherals and for the user RAM.

<table>
<thead>
<tr>
<th>DEVICE</th>
<th>TYPE</th>
<th>ADDRESSES</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

162
MP 11  DATA  17F0 - 17 F7
D/A  ( channels 0-7 )
MP 21  CONTROL  1FF2, 1FF3
A/D  ( channels 0-3 )  1FF6, 1FF7
      DATA  1FF0, 1FF1
      ( channels 0-3 )  1FF4, 1FF5
USER  RAM  0000 - 0FFF
       ( continuos )

TABLE 1 - Memory Map for Peripherals & User RAM

To use the MP 11 A/D converter two Load accumulator instructions are needed. The first starts the conversion. After at least 25 usec, the second load will be actually load the accumulator with the digital value of the input. If an input is greater than +5 volts or less than -5 volts the digital number will be the same as for +5 and -5 respectively.

To use the MP 21 D/A a store instruction is all that is needed.

D. STEPPING MOTOR TRANSLATOR
The translator is the STM 1800 CV made by Superior Electric. It accepts pulses at a rate of from 20 to 2000 pulses per second. This corresponds to a pulse separation of from 50 msec (20 pulses/sec) to 500 usec (2000 pulses per sec). Each pulse must be at least 30 usec wide. The translator will step the motor on the leading edge of a negative-going pulse or the trailing edge of a positive-going pulse. The high level is +10 volts. The low level is -10 volts. Each value can be between 8-10 volts. There are two separate inputs to the translator, one for clockwise rotation of the motor and one for counter-clockwise rotation. There is also a common (ground) input.

SPEED CONTROL BY LINEAR INPUT VOLTAGE

A. GENERAL TIMING CONSIDERATION
A series of pulses is generated using the signal processor by programming a time delay between a program segment which loads the analog output device in a way to produce one pulse. Both program segments, the time delay and the pulse output, are repeated as long as needed. The pulses drive a stepping motor. To accelerate (or decelerate) the
motor the time delay between pulses is decreased (or increased).

Figure 2 shows the timing involved in putting out a series of pulses. (Note that the pulses are positive pulses instead of the negative pulses required by the stepping motor translator. This is only for ease of reading.) PI is the time between the end of one pulse and the start of the next pulse. It is an abbreviation of pulses interval. MI is the minimum PI. This interval corresponds to the fastest pulse rate allowed by the stepping motor translator. AMI is the absolute minimum interval. AMI represents the execution time for the program itself. The smallest MI can be is AMI. Delay is the extra time (besides MI) which is needed to match the pulse rate to the input voltage. The actual time between pulses is labeled pulse Delay. This is the time between the start of one pulse and the start of the next pulse. The smallest pulse Delay is the minimum interval MI. Pulse Delay has a range of from fifty milliseconds (50 ms) to five hundred microseconds (500 usec) for the stepping motor translator in the Servo lab. The pulse Delay is equal to the PI plus the pulse width. Since the pulse width is constant, the pulse Delay varies as PI varies. The programmed delay between pulses described in the preceding paragraph delays an amount PI.

**FIG 2 - Program Timing**

164
B. THE METHOD USED

The method chosen was to use a table of counters and to access the counters by using the input from the A/D converter as an offset address. For example, a zero input would give as the counter at the beginning of the table.

The table does not contain an address (containing a counter) for every possible input. If this were true the table would need to be 127 bytes long. The inputs are classified as slow speed or high speed. An input (absolute value in case of minus voltages) less than 10 hex is considered low speed. All other inputs are classified as high speed.

In the slow speed range an input will give the exact offset. For example, suppose the table starts at location 0100 hex. If the input is 09 the address of the desired counter is location 0109.

The high speed range is divided into three parts. An input, XPI between 11 and 1F hex (inclusive) will give directly the address of the counter. These addresses are 0111 to 011F hex. An input from 20 to 40 hex will give an address between 0120 to 012F. An input greater than 40 will give an address of 0130 to 0140. This is summarized below. The equation refers to the algebraic manipulation on the input to get the offset address. The table starts at 0100.

<table>
<thead>
<tr>
<th>VALUE IF INPUT XPI</th>
<th>EQUATION</th>
</tr>
</thead>
<tbody>
<tr>
<td>$00_{16} \leq XPI \leq 10_{16}$</td>
<td>$Address = 0100_{16} + x$ Low speed</td>
</tr>
<tr>
<td>$10_{16} &lt; XPI &lt; 20_{16}$</td>
<td>$Address = 0100_{16} + x$ High speed</td>
</tr>
<tr>
<td>$20_{16} \leq XPI &lt; 40_{16}$</td>
<td>$Address = 0120_{16} + \frac{(x - 20_{16})}{2_{16}}$ High speed</td>
</tr>
<tr>
<td>$40_{16} \leq XPI$</td>
<td>$Address = 0130_{16} + \frac{(x - 40_{16})}{4_{16}}$ High speed</td>
</tr>
</tbody>
</table>

TABLE 2 - CALCULATION OF ADDRESS OF COUNTER
TABLE 3 - RELATIONSHIP BETWEEN LOOP COUNTERS AND INPUT

The reasons this method was chosen can be seen in TABLE 3. The time delay, the delay between pulses, is based on an MI loop. The shortest delay between pulses occurs when the motor is rotating at a maximum speed. The corresponds to an interval between pulses of MI. Therefore, longer delay can be realized by performing an MI delay many times. In TABLE 3 the number of MI loops required for a certain input is K.

The reason different speed ranges were needed is that the number of steps between inputs (XPI) is large for an input greater than 10 hex. this would require a large table of counters.

The MI timing loop is below.

```
LDAB    K
DECB
X1
LDAA    N
# of MI'S performed
X2
DECA
BNE X2
DECB
BGT X1
```
The number of cycles for the MI loop execution is

$$MI_n = 3 + N \cdot G.$$  

For the stepping motor translator used the MI is 500 usec (2000 pulses per second), with the MPU clock set at 614.4 KHz this is an $MI_n$ of 307 (base 10) cycles, therefore, $N$ should be fifty base 10 or 32 base 16.

This has been a short explanation of the method used in the program for determining pulse intervals. This method is best understood by looking at the program itself.

C. THE PROGRAM

1. GENERAL

The table of counters is listed in Figure 3. Notice that, for slow speed ($00 \leq XPI \leq 10$), the addresses match directly with the XPI and that the corresponding counter for an XPI agrees with those shown as K in TABLE 3. Also, notice that there is a counter for each XPI - each step is taken care of. For example, between $XPI = 08$ and $XPI = 10$ there are eight steps. In the table, between $XPI = 08$ and $XPI = 10$ there are also eight steps.

In the high speeds range notice that each counter K corresponds to the counter K in TABLE 3 for the high XPI of the range. For example, if the XPI is 24 the address (see TABLE 2) of this counter is:

$$\frac{24 - 20}{2} + 0120 = 0122$$

This gives a counter K of 2 which is the counter in TABLE 3 for an input XPI of 40.

The table of counters for high speeds also contains a number DX. This is used to time the extra delay needed because a value of XPI does not exactly equal 20, 40, or 80. DX is the number of L loops needed to make the delay proportional to the input.

For example, suppose $XPI = 24$. The address as calculated is 0122. This gives a K of 2 and a DX of E. The total delay between pulses is then

$$\text{Delay} = K \cdot MI + DX \cdot L = 2 \cdot MI + E \cdot L$$

K-MI is the delay for the highest input in a range.
DX · L is the extra delay for the difference between the actual input XPI and the highest input in a range. (See TABLE 3.)

The Ls are determined as follows:

Range 1: \( K = 08 - 04 = 04 \)

\[# steps = 10_{16} \]

\[ L = \left(\frac{04}{10}\right)^n = \frac{N}{4} \]

Range 2: \( K = 04 - 02 = 02 \)

\[# steps = 20_{16} \]

\[# address steps (different \# counters) = 10_{16} \]

\[ L = \left(\frac{02}{10}\right)^n = \frac{N}{8} \]

Range 3: \( K = 02 - 01 = 01 \)

\[# steps = 40_{16} \]

\[# address steps = 10_{16} \]

\[ L = \left(\frac{01}{10}\right)^n = \frac{N}{10} \]

Each range must increment or decrement the input differently to account for only 10\(_{16}\) addresses (counters) per range. Therefore, in range 1 the input is incremented or decremented by 1 · (as in the low speed range). In range 2 the factor is 2 and in Range 3 the factor is 4. The input is incremented for acceleration and decremented for deceleration.
<table>
<thead>
<tr>
<th>ADDRESS</th>
<th>CONTENTS</th>
<th>XPI</th>
<th>ADDRESS</th>
<th>CONTENTS</th>
<th>DX</th>
<th>K</th>
</tr>
</thead>
<tbody>
<tr>
<td>0100</td>
<td>80</td>
<td>01</td>
<td>0121</td>
<td>7A</td>
<td>F</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>40</td>
<td>02</td>
<td>2</td>
<td>72</td>
<td>E</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>30</td>
<td>03</td>
<td>3</td>
<td>6A</td>
<td>D</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>20</td>
<td>04</td>
<td>4</td>
<td>62</td>
<td>C</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>1C</td>
<td>05</td>
<td>5</td>
<td>5A</td>
<td>B</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>18</td>
<td>06</td>
<td>6</td>
<td>52</td>
<td>A</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>14</td>
<td>07</td>
<td>7</td>
<td>4A</td>
<td>9</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>10</td>
<td>08</td>
<td>8</td>
<td>42</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>0F</td>
<td>09</td>
<td>9</td>
<td>3A</td>
<td>7</td>
<td>2</td>
</tr>
<tr>
<td>9</td>
<td>0E</td>
<td>0A</td>
<td>A</td>
<td>32</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>A</td>
<td>0D</td>
<td>0B</td>
<td>B</td>
<td>2A</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>B</td>
<td>0C</td>
<td>0C</td>
<td>C</td>
<td>22</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>C</td>
<td>0B</td>
<td>0D</td>
<td>D</td>
<td>1A</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>D</td>
<td>0A</td>
<td>0E</td>
<td>E</td>
<td>12</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>E</td>
<td>09</td>
<td>0F</td>
<td>F</td>
<td>0A</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>F</td>
<td>08</td>
<td>10</td>
<td>0130</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0110</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0111</td>
<td>7C</td>
<td>F</td>
<td>4</td>
<td>0131</td>
<td>79</td>
<td>F</td>
</tr>
<tr>
<td>2</td>
<td>74</td>
<td>E</td>
<td>4</td>
<td>2</td>
<td>71</td>
<td>E</td>
</tr>
<tr>
<td>3</td>
<td>6C</td>
<td>D</td>
<td>4</td>
<td>3</td>
<td>69</td>
<td>D</td>
</tr>
<tr>
<td>4</td>
<td>64</td>
<td>C</td>
<td>4</td>
<td>4</td>
<td>61</td>
<td>C</td>
</tr>
<tr>
<td>5</td>
<td>5C</td>
<td>B</td>
<td>4</td>
<td>5</td>
<td>59</td>
<td>B</td>
</tr>
<tr>
<td>6</td>
<td>54</td>
<td>A</td>
<td>4</td>
<td>6</td>
<td>51</td>
<td>A</td>
</tr>
<tr>
<td>7</td>
<td>4C</td>
<td>9</td>
<td>4</td>
<td>7</td>
<td>49</td>
<td>9</td>
</tr>
<tr>
<td>8</td>
<td>44</td>
<td>8</td>
<td>4</td>
<td>8</td>
<td>41</td>
<td>8</td>
</tr>
<tr>
<td>9</td>
<td>3C</td>
<td>7</td>
<td>4</td>
<td>9</td>
<td>39</td>
<td>7</td>
</tr>
<tr>
<td>A</td>
<td>34</td>
<td>6</td>
<td>4</td>
<td>A</td>
<td>31</td>
<td>6</td>
</tr>
<tr>
<td>B</td>
<td>2C</td>
<td>5</td>
<td>4</td>
<td>B</td>
<td>29</td>
<td>5</td>
</tr>
<tr>
<td>C</td>
<td>24</td>
<td>4</td>
<td>4</td>
<td>C</td>
<td>21</td>
<td>4</td>
</tr>
<tr>
<td>D</td>
<td>1C</td>
<td>3</td>
<td>4</td>
<td>D</td>
<td>19</td>
<td>3</td>
</tr>
<tr>
<td>E</td>
<td>14</td>
<td>2</td>
<td>4</td>
<td>E</td>
<td>11</td>
<td>2</td>
</tr>
<tr>
<td>F</td>
<td>0C</td>
<td>1</td>
<td>4</td>
<td>F</td>
<td>09</td>
<td>1</td>
</tr>
<tr>
<td>0120</td>
<td>04</td>
<td>0</td>
<td>0140</td>
<td>01</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

**CONTENTS:**

**FIGURE 3 - COUNTER TABLE**

169
2 FLOW CHARTS AND LISTINGS

Figures 4 - 11 are the flow charts for the various program segments. Labels for these charts match those labels in the program listing. Output Up refers to a positive-going pulse. In reality the pulses are negative-going.

Directly after each flowchart is the listing of the corresponding program segment. The left four numbers are the address of the instruction as it is loaded into the tape (cassette). The next two numbers are the op-code of the instruction. Any other numbers on a line are branch offsets, immediate data, or an address. The TLOOP Program segment does not have a flowchart. It has been explained in the previous section. The following program segments constitute the program.

INITIALIZATION - Sets up L Values, Clears some memory.

STRUN - Main program, checks inputs & gets loop counters.

Z - Zero Speed routine, will change output channel if necessary.

GN - Gets a new value from A/D converter, Sets change channel flag.

TLOOP - Pulse delay timer.

PULSE - Outputs a pulse 52 - (84.6 usec) wide. Also counts down pulses using counter & counter flag CTRF.

TC - Time compensation for constant speed.

TLOOP & TC are presented on the next page.

On the page after that is a listing of low memory addresses. Figure 4 shows the general program flow, The program segments follow that.

NOTE: XS 2 in the program listings & flowcharts is the same as XPI on the pages before this page in this report.
TLOOP

02EF 7D 000F  TST: HF  #SPEED?
   26 0C  BNE CONT
   68 0B  LDAB K
   5A  DECB
   96 0A  T3 LDAA N
   4A   T2 DECA
   26 FD  BNE T2
   5A  DECB
   2E FB  BGT T3
   39   RTS

0300 D4 0B CONT LDAB K  #HIGH SPEED.
   5A  DECB
   27 08  BEQ T4  #IF K=1 SKIP FIRST PAIR OF LOOPS.
   96 0A  T6 LDAA N
   4A   T5 DECA
   2E FD  BGT T5
   5A  DECB
   2E FB  BNE T6
   86 09  T4 LDAB DX
   5A   T9 DECB
   2D 07  BMI T7
   96 0E  LDAA L
   4A   T8 DECA
   2E FD  BGT T8
   20 F6  BRA T9

03A 39  T7 RTS

PAGE ZERO ADDRESSES

0000  CTRF  #CONSTANT N.
0001  N  #PRESENT VALUE OF INPUT FROM A/D CHIP.
0002  XP 4  X$1 5  X$2
0003  6  TP  #COUNTER FOR COMPENSATION LOOP.
0004  K 7  LOOP COUNTER.
0005  DX 8  HIGH SPEED LOOP COUNTER.
0006  N 9  #TEMPORARY N.
0007  A  A
0008  B  B
0009  C  C
000A  D  D  #HIGH SPEED LOOP COUNTER.
000B  E  E  #SPEED FLAG. HF=0 INDICATES LOW SPEED.
0010  L  F  #CHANGE CHANNEL FLAG. CWI=1 CHANGE CHANNEL
0011  CMI 1  DESIRED CHANNEL. =0 CW. =1 CCW.
0012  CCW 2  #PRESENT CHANNEL. =0 CW. =1 CCW.
0013  CCWP 3  TPMP1 4  TPMP2 5  #LOCATION WHERE ADDRESS OF LOOP
0014  TMP1 6  TMP2 7  #COUNTER FROM TABLE IS STORED.
0015  FP 8  #OUTPUT
0016  F0 9  CHANNEL
0017  FF 0  PULSE COUNTER FLAG.
0018  CTRF  A  PULSE COUNTER MSR.
0019  C2  B  PULSE COUNTER MSR.
001A  C1  C  PULSE COUNTER LSR.
001B  CO  D

TC

0370 86 07  LDAA #$07
   4A  TC1 DECA
   2E FD  BGT TC1
   39  RTS
FIGURE 4 - PROGRAM FLOW

Names in () are actual program segment names,
Numbers in [] are the number of cycles
for the programs within the boxes,

NOTE: STRUN is 85 cycles long in execution.
FIGURE 6 - Main Program STRUN

This is similar to the block diagram of Figure 4.
FIGURE 7 - Acceleration Routine
FIGURE 8 = Deceleration Routine

Labels are in Acceleration routine, See FIG.
START RUN

0190 96 03  LDA A, XP  ;
91 05  CMPA XS2  ; (XP - XS2)?
26 12  BNE A
7D 0005  TST XS2  ;XP=XS2. CONSTANT SPEED.
26 03  BNE A1  ;SPEED IS ZERO.
7E 0260  JMP Z  ;GO TO ZERO ROUTINE.
01  NOP  ;
01  NOP  ;
D7 05  STAB XS2  ;
BD 0370  JSR TC  ;TIME COMPENSATION FOR PROGRAM
7E 023F  JMP ADDR  ;
2D 03  A.BLT A2  ;ACCELERATION.
7E 0212  JMP DEC  ;DECELERATION.
01  A2  ;
01  NOP  ;
01  INCA  ;XP=XP+1
01  NOP  ;
01  NOP  ;
81 10  CMPA #$10  ;SPEED?
2F 03  BLE A3  ;
7E 01B7  JMP C  ;GO TO HIGH SPEED RTN.
7F 000F A3 CLR HF  ;LOW SPEED.
01  D  NOP  ;
01  NOP  ;
01  NOP  ;
01  NOP  ;
01  NOP  ;
01  NOP  ;
97 03  STAA XP  ;
97 15  D1 STAA TMP2  ;TMP2=XP
01  NOP  ;
01  NOP  ;
01  NOP  ;
01  NOP  ;
01  NOP  ;
01  NOP  ;
01  NOP  ;
7E 023F  JMP ADDR  ;

START RUN (CONTINUED)

01D7 C6 01  C LDA B #$01  ;LOW SPEED.
D7 0F  STAB HF  ;
01 21  CMPA #$20  ;
2E 07  BGT E1  ;
D6 0E  LDA A  ;IL=(N/4)
D7 0E  STAB L  ;
7E 01CB  JMP D1  ;
01  E1  INCA  ;XP >= 20. XP=XP+1.
81 40  CMPA #$40  ;
2E 13  BGT G1  ;
D6 0C  LDA B  ;$20 <= XP <= 40.
D7 0E  STAB L  ;IL=(N/8)
01  NOP  ;
01  NOP  ;
01  NOP  ;
97 03  STAA XP  ;TMP2=((XP-20)/2 + 20)  ;
80 20  SUSA #$20  ;
44  LSRA  ;
88 20  ADDA #$20  ;
97 15  STAA TMP2  ;
7E 023F  JMP ADDR  ;
01  C1  NOP  ;XP>40.  
01  INCA  ;XP=XP+2
01  INCA  ;
0201 97 03  H STAA XP  ;
86 0D  LDA B  ;
D7 0E  STAB L  ;IL=(N/10)
80 40  SUSA #$40  ;TMP2=((XP-40)/4 + 30)
44  LSRA  ;
44  LSRA  ;
8D 30  ADDA #$30  ;
97 15  STAA TMP2  ;
7E 023F  JMP ADDR  ;
4A  DEC DCA  ;DECELERATION. XP=XP-1.
26 05  BNE H1  ;
97 03  STAA XP  ;
7E 0280  JMP Z  ;
01 10  H1 CMPA #$10  ;
2F 03  BNE H2  ;
7E 0227  JMP H3  ;
7F 000F H2 CLR HF  ;
7E 01BE  JMP D  ;
0227 D6 01  H3 LDA B #$01  ;
D7 0F  STAB HF  ;
Start Run (Continued)

01 20 CMPA B#20
    2E 03 MOV X2
    7E 01BF JMP E
01 X2 JPC
01 40 CMPX B#40
    3E 03 DLE X3
01 7E 00EM JMP C
01 X3 JPCA
01 BCA

023F 0E 14 ADD X LDA X,TP1
    16 00 LDAA B#0,X
    76 000F TST HF
    3E 0A MVC AB1
01 NOP
01 NOP
01 NOP
01 NOP
01 NOP

97 08 STAA K
20 09 BRA GN
84 07 ADI ADAA B#07
97 08 STAA K
66 00 LDAA B#0,X
44 LSR A
44 LSR A

97 09 STAA DX
80 02EF JSR TLOOP
80 0326 JK PULSE
0263 7E 027C JMP GN

Figure 9 - Gain Routine

Reads A/D, checks output channel, converts input to absolute value, checks for 80 input & sets it to
FIGURE 10 - ZERO SPEED ROUTINE
FIGURE 11 - Pulse Output Routine

```
02B0 66 17F0  Z LDAA A/D  ; Zero speed.
06 77F  LDAA 167F  ; Output down.
DE 16  LDX CHANNEL  ;
A7 00  STAA 40x  ;
96 11  LDAA CWI  ;
27 06  BEQ Z1  ;
60 02D0  JSR CHN  ;
7F 0011  CLR CWI  ; Clear flag.
01  Z1 NOP  ; CWI=0, no change.
01  NOP  ;
97 05  STAA X52  ; X52=X51
96 12  LDAA CCW  ;
97 13  STAA CCNP  ;
02CD 7E 0380  JMP CEAT  ; Go to gn but first eat some z.
02D0 96 17  CHN LDAA #17  ;
01 F0  CHPA #17F0  ;
26 05  BNE CC1  ; Current channel = 1FF0?
4C  INCA  ; Yes, change it.
97 17  STAA #17  ;
20 04  BRA CC2  ;
86 F0  CC1 LDAA #F0  ; No, change it.
97 17  STAA #17  ;
02DF 39 CC2 RTS  ;
0380 86 07  CEAT LDAA #07  ; Cycle eater.
4A CE2 DECA  ;
2E FD  BGT CE2  ;
7E 0263 JMP GN  ; Get next ad value.
```

STOP

Yes

COUNTER = COUNTER - 1
(3 bytes)

52 cycle delay

Yes

STOP

COUNTER = 0

No

GaNa

Set up A/D for Future Read

CINR=0

Yes

No

Pulse Output

FUSE LOOP

0320 B6 17F0    LDA A AD
B6 FF    LDA A #FF
DE 16    LDX CHANNEL
A7 06    STA A #0X
7D 0019   TST CTFX
26 07    BNE CN3
B6 06    LDA A #06
4A CN1    DECA
26 FB    BNE CN1
20 15    SRA CN2
96 1C    CN3 LDA A CO
80 01    SUBA #101
97 1C    STA A CO
96 18    LDA A C1
B2 00    SBC #100
97 10    STA A C1
96 1A    LDA A C2
B2 00    SBC #100
97 1A    STA A C2
24 61    PCC CN2
3F SWI    COUNT = 0, STOP.
86 7F    CN2 LDA A #7F
DE 16    LDAA CHANNEL
A7 00    STA A #0X
39 RTS    OUTPUT DOWN.

;SET UP AD FOR FUTURE READ...
;OUTPUT UP.

; CYCLE EATER.
; NO COUNT, CYCLE EATER.

;FLAG WAS ZERO.
;GET COUNTER.

; 3 BYTE SUBTRACTION.

180
D. Using THE PROGRAM

To use the program first load it into memory from the cassette tape. The tape number starting at 00 corresponds to the beginning of the RAM memory. The tape should run to about 19 before the program is loaded. Memory is loaded from 0000 to 0400.

To first run the program load N into its page zero location. Also, enter the permanent counter flag CTRF (0001) and the desired counter. The first time the program is run must include the INITIALIZATION program. After this program is run, the program can be rerun by running STRUN. The INITIALIZATION starts at location 0150. STRUN starts at 0190.

Be sure to connect a voltage source to analog input channel 0. Also, connect the analog output channels 0 and 1 to the stepping motor translator. The channels are not initialized so a specific channel does not correspond with a certain direction.

**POSITION CONTROL CONSIDERATION**

As mentioned in the previous section, all the stepping motor need some ramp pulse interval to speed up itself. Consequently, due to the inertia we also have to use some deceleration technique to make it stop at the precise angle position. The pulse interval from the beginning to the end of the rotation should be distributed as follows:

![Diagram showing pulse interval](image)

Number of steps is calculated by the users need

**Figure 11. Position control pulse interval.**

Before making any effort, we find two conclusions from what we got in the previous section. First, we can accelerate or decelerate the motor properly with 64 steps ramp interval. Second, stepping motor when used to control position the total number of steps should be divided into three parts, those are: 1) acceleration steps. 2) maximum speed steps. 3) deceleration steps. Moreover, 1) and 3) can be equal. Thus, we need to consider the problem only in two cases:
Fig 12 Acceleration and Deceleration -
Counter consideration

Fig 14 CONTROL CODE decoder circuit
Fig 13 Flow chart for counter choice
CASE I: When present step counter $PC \leq 64$, (see Fig. 12) the speed counter address $N (0 \sim 64)$ must choose $PC$ when $PC < RC$ ($RC$ is the rest step counter), otherwise choose $RC$.

CASE II: When present step counter $PC > 64$, we must set $N$ equal to maximum speed address 64 before $PC$ reach 64, and after $RC$ come down to less than 64 we just set $N=RC$.

Now, we draw the flow chart for the stepping motor position control in Fig 13. Applying this flow chart we can easily construct an interface for the motor as done in the next section.

**POSITION CONTROL INTERFACE DESIGN**

A. Instruction decoder.

Since we want to run several stepping motor in the same time parallely through interface adapter P1AA and P1AB. We have to use some control operation code as follows:

```
Instruction code    nemonic
0000xxxxaaa        CLR ;RESET THE WHOLE SYSTEM.
001xxxxaaa         HLT ;HALT MOTOR.
010xxxxaaa         CTRN ;CONTINUOUS RUNNING.
011xxxxaaa         CPM ;MANUAL CONTROL.
100xxxxaaa         ENI ;ENABLE INTERRUPT.
101xxxxaaa         LD C ;LOAD COUNTER.
110xxxxaaa         ENCH ;ENABLE CHECK.
1110xxxxaaa        GO+ ;START RUN + DIRECTION.
1111xxxxaaa        GO- ;START RUN - DIRECTION.
```

In Figure 14 we construct the control code decoder by CD4099 and in Figure 15, we use DM7485 as the main comparators to select the right counter for speed counter address. Here, we need a modulo 6 counter to control the operation sequence. See Figure 16.

B. Speed counter memory and K.L determination

In Figure 16 we use MCM6810A RAM to store the counter table. Note that all the number we used in here are decade
Fig. 15 Comparator & Pulse Counter Circuit.

The diagram shows a complex electronic circuit with various components and connections. The labels and symbols suggest a detailed circuit analysis, possibly related to a specific application or system.
1. All IC not specified use TTL by NSC
2. Output to motor translator need a level change to +10V which not included here

**Figure 16** SEQUENCE A TIME INTERNAL COUNTER

- **K Counter**
- **L Counter**
- **Sequence Counter**
- **Clocks CK3, CK4, CK5**
even ACC. Therefore, inside the RAM we just discard 0A 0F addresses. Also note that, since we count the time interval only by two sets of cascade decade counter, then we have to change the counter table bits as when \( N < 16 \), bits 0 - 7 are K count and when \( N \geq 16 \) bits 7 - 4 are K count while bits 3 - 0 are L count. See Figure 17.

**CONTROL PROGRAM**

The control program are much simple than before. Since after initialization, each motor will run by a single "GO" instruction. A short example is list right after.

*THE FOLLOWING PROGRAM LOAD STEP COUNTER WITH 123456 STEPS AND THEN START THE STEPPING MOTOR TO RUN WITH ITS MAXIMUM SPEED TO THE POSITION WE SET*

```
0400 CE 8004 LDX #$8004 ;INITIALIZATION. SET PIAA & PIAB AS OUTPUT.
  6F 03 CLR CRB
  6F 01 CLR CRA
  86 FF LDAA #$FF
 A7 00 STAA #$FF
 A7 02 STAA DDRA
  86 04 STAA DDRB
 A7 03 LLDAA #$04
 A7 01 STAA CRA
  0411 4F STAA CRB
 A7 00 CLRA ;RESET ALL REGISTERS IN MOTOR "1" INTERFACE.
  86 04 STAA PIAA
 A7 02 LDAA #$64 ;USE 64 STEPS TO ACCELERATE
  86 A8 STAA PIAB
 A7 00 LDAA (01101000)_B ;SENT LOAD ACC INSTRUCTION.
  86 12 STAA PIAA
  86 12 LDAA #$12 ;LOAD RCH WITH 12
  0420 A7 02 STAA PIAB
  86 A0 LDAA (10100000)_B ;RCH LOADED
 A7 00 STAA PIAA
  86 34 LDAA #$34 ;LOAD RCM WITH 34
 A7 02 STAA PIAB
  86 B0 LDAA (10110000)_B ;RCM LOADED
 A7 00 STAA PIAA
  86 56 LDAA #$56 ;LOAD RCL WITH 56
```
<table>
<thead>
<tr>
<th>n</th>
<th>K</th>
<th>L</th>
<th>n</th>
<th>K</th>
<th>L</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>80</td>
<td>0</td>
<td>33</td>
<td>03</td>
<td>14</td>
</tr>
<tr>
<td>2</td>
<td>40</td>
<td>0</td>
<td>34</td>
<td>03</td>
<td>12</td>
</tr>
<tr>
<td>3</td>
<td>30</td>
<td>0</td>
<td>35</td>
<td>03</td>
<td>10</td>
</tr>
<tr>
<td>4</td>
<td>20</td>
<td>0</td>
<td>36</td>
<td>03</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>1C</td>
<td>0</td>
<td>37</td>
<td>03</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>18</td>
<td>0</td>
<td>38</td>
<td>03</td>
<td>4</td>
</tr>
<tr>
<td>7</td>
<td>14</td>
<td>0</td>
<td>39</td>
<td>03</td>
<td>2</td>
</tr>
<tr>
<td>8</td>
<td>10</td>
<td>0</td>
<td>40</td>
<td>03</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>0F</td>
<td>0</td>
<td>41</td>
<td>02</td>
<td>14</td>
</tr>
<tr>
<td>10</td>
<td>0E</td>
<td>0</td>
<td>42</td>
<td>02</td>
<td>12</td>
</tr>
<tr>
<td>11</td>
<td>0D</td>
<td>0</td>
<td>43</td>
<td>02</td>
<td>10</td>
</tr>
<tr>
<td>12</td>
<td>0C</td>
<td>0</td>
<td>44</td>
<td>02</td>
<td>8</td>
</tr>
<tr>
<td>13</td>
<td>0B</td>
<td>0</td>
<td>45</td>
<td>02</td>
<td>6</td>
</tr>
<tr>
<td>14</td>
<td>0A</td>
<td>0</td>
<td>46</td>
<td>02</td>
<td>4</td>
</tr>
<tr>
<td>15</td>
<td>09</td>
<td>0</td>
<td>47</td>
<td>02</td>
<td>2</td>
</tr>
<tr>
<td>16</td>
<td>08</td>
<td>0</td>
<td>48</td>
<td>02</td>
<td>0</td>
</tr>
<tr>
<td>17</td>
<td>07</td>
<td>12</td>
<td>49</td>
<td>01</td>
<td>15</td>
</tr>
<tr>
<td>18</td>
<td>07</td>
<td>8</td>
<td>50</td>
<td>01</td>
<td>14</td>
</tr>
<tr>
<td>19</td>
<td>07</td>
<td>4</td>
<td>51</td>
<td>01</td>
<td>13</td>
</tr>
<tr>
<td>20</td>
<td>07</td>
<td>0</td>
<td>52</td>
<td>01</td>
<td>12</td>
</tr>
<tr>
<td>21</td>
<td>06</td>
<td>12</td>
<td>53</td>
<td>01</td>
<td>11</td>
</tr>
<tr>
<td>22</td>
<td>06</td>
<td>8</td>
<td>54</td>
<td>01</td>
<td>10</td>
</tr>
<tr>
<td>23</td>
<td>06</td>
<td>4</td>
<td>55</td>
<td>01</td>
<td>9</td>
</tr>
<tr>
<td>24</td>
<td>06</td>
<td>0</td>
<td>56</td>
<td>01</td>
<td>8</td>
</tr>
<tr>
<td>25</td>
<td>05</td>
<td>12</td>
<td>57</td>
<td>01</td>
<td>7</td>
</tr>
<tr>
<td>26</td>
<td>05</td>
<td>8</td>
<td>58</td>
<td>01</td>
<td>6</td>
</tr>
<tr>
<td>27</td>
<td>05</td>
<td>4</td>
<td>59</td>
<td>01</td>
<td>5</td>
</tr>
<tr>
<td>28</td>
<td>05</td>
<td>0</td>
<td>60</td>
<td>01</td>
<td>4</td>
</tr>
<tr>
<td>29</td>
<td>04</td>
<td>12</td>
<td>61</td>
<td>01</td>
<td>3</td>
</tr>
<tr>
<td>30</td>
<td>04</td>
<td>8</td>
<td>62</td>
<td>01</td>
<td>2</td>
</tr>
<tr>
<td>31</td>
<td>04</td>
<td>4</td>
<td>63</td>
<td>01</td>
<td>1</td>
</tr>
<tr>
<td>32</td>
<td>04</td>
<td>0</td>
<td>64</td>
<td>01</td>
<td>0</td>
</tr>
</tbody>
</table>
RESULTS

The program was loaded in the digital processor and tested using an analog voltage source. The program was started and the potentiometer was varied. The motor responded very well when the voltage was between -5 and +5 volts. Outside this range the motor will run with its maximum speed at each direction. From this test we found that our stepping motor can never run with a speed more than 800 steps/second without acceration. If we suddenly apply a voltage pulse input with 900 steps/sec it were buzzing and vibrating at the initial position.

In the second part we set the counter and direction code with various number and run it. We found the motor will stop at the exact position as we predict, so, this part is also perfect for a position control.

REFERENCES

(1) "Theory & Application of step motors", B.C. KUO
    Department of E.E. University of Illinois at Urbana
    Champaign Urbana, Illinois.
(2) "M8800 Microcomputer System Design Data" Motorola
    I.N.C.
(3) "National Semiconductor Data Book" N.S.C.
(4) "BURR BROWN GUIDE TO ANALOG INPUT MICRO
    PERIPHERAL UNIT", BURR BROWN.
(5) "BURR-BROWN GUIDE TO ANALOG OUTPUT MICRO
    PERIPHERAL UNIT", BURR-BROWN.
(6) "SLO-SYN SYNCHRONOUS/ STEPPING MOTORS AND MOTOR
CONTROLS", Superior Electric.
(7) "MOTOROLA EVALUATION KIT MANUAL", MOTOROLA.
摘要

應用M 6800微處理機作步級馬達之加減速與位置控制之研究

黃國恩

本文係研究如何以微處理機直接產生週期可調之脈波列來控制步級馬達之加減速，以突破馬達之諧振週期而達預定之轉速。文之前段係以外加可變電壓為輸入信號，指揮步級馬達之正反轉與變速。第二段則研究以預儲式分段加減速之間隔後求出預定前進位置（N）與所需變速之關係以得到最佳之位置控制。第三段則嘗試以硬體之結構來完成此加減速與位置控制之裝置。