2.7
3/7/79
2.4 to 2.15

MASTER

BDX-613-2072 (Rev.)

MICROPROCESSOR CIRCUIT MONITOR

By D. J. Stevens

Published February 1979

Final Report

Prepared for the United States Department of Energy
Under Contract Number DE-AC04-76-DP00613.

**Kansas City
Division**

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency Thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Department of Energy, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

Printed in the United States of America

Available From the National Technical Information Service, U.S. Department of Commerce, 5285 Port Royal Road, Springfield, Virginia 22161.

Price:	Microfiche	\$3.00
	Paper Copy	\$4.00

MICROPROCESSOR CIRCUIT MONITOR

By D. J. Stevens

Published February 1979

Final Report
D. J. Stevens, Project Leader

Project Team:
P. F. Arnold
R. Biase
R. C. Harnden

NOTICE

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Department of Energy, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product or process disclosed, or represents that it has been tested or reviewed for accuracy or completeness.

Communications Services

**Kansas City
Division**

MICROPROCESSOR CIRCUIT MONITOR

BDX-613-2072 (Rev.), Final Report, Published February 1979

Prepared by D. J. Stevens

A microprocessor based diagnostic system was developed to monitor a microprocessor circuit while it is operating. The system determines if selected memory locations are addressed in the correct sequence and records time to the nearest 0.001 second when each selected memory location is addressed.

WM-TR12/a

This report was prepared as an account of work sponsored by the United States Government. Neither the United States, nor the United States Department of Energy, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, expressed or implied or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights.

The Bendix Corporation
Kansas City Division
P. O. Box 1159
Kansas City, Missouri 64141

A prime contractor with the United States
Department of Energy under Contract Number
DE-AC04-76-DP00613

CONTENTS

Section	Page
SUMMARY.	5
DISCUSSION	6
SCOPE AND PURPOSE.	6
ACTIVITY	6
<u>Approach</u>	6
<u>System Development</u>	9
ACCOMPLISHMENTS.	16
FUTURE WORK.	19
APPENDIX. PROGRAM LISTING	20
DISTRIBUTION	25

ILLUSTRATIONS

Figure		Page
1	Block Diagram of Microprocessor Monitor . . .	7
2	Electrical Connections to CPU and SMI	8
3	Address Comparison Circuit Schematic.	11
4	Microprocessor Monitor Connected to Electronic Programmer Printed Wiring Assembly (P-102114)	12
5	Static Memory Interface, Down Counter, and Comparison Circuits (P-102113).	13
6	Clock Reference Frequency Circuit Modification.	14
7	Flow Chart of Main Program.	15
8	PSU Interrupt Service Routine	16
9	Subroutine Dump	17
10	Test Printout and Stored Information.	18

SUMMARY

A new laboratory capability has been developed which will be used to support some microprocessor based products such as an electronic programmer. The monitor system allows the user to select samples from the instruction set of the microprocessor product under test and verify its progress through those instructions during a test run. The monitor records the time when each selected instruction is addressed and prints out a record of address and time data when a run is complete. These capabilities provide a significant diagnostic tool for investigating a microprocessor malfunction.

As an example, the monitor system was successfully used to locate and identify a change that had been made to the instruction set of an electronic programmer.

This monitor system is designed to test specific product. The concept, however, may be applied to other microprocessor based products.

DISCUSSION

SCOPE AND PURPOSE

This work was done to develop laboratory methods for evaluating microprocessor based circuits. Because of the predictable, sequential manner in which instructions are fetched from memory with a known program and correct instruction execution, the technique of monitoring addresses used for calling up information was selected. With the ability to determine if instructions of interest are fetched and the times when these instructions are fetched, a useful diagnostic tool would be made available.

ACTIVITY

Work began with the assembly of the comparing circuits. When these were found to function as intended, software instructions were written to store time information to the nearest 0.01 second. During the initial phase a free running multivibrator was used as a time base and no attempt was made to generate precise time pulse intervals. The next step involved refinement of the time base to improve precision and altering software instructions to store time information to the nearest 0.001 second. Next, the printout routine was added to the software instructions so that data could be readily interpreted.

The monitor was then exercised with product and diagnostic data was analyzed.

Approach

A microprocessor design evaluation kit with 1024 bytes of program or data storage capacity was used to perform most of the required functions (Figure 1). The kit was used to sequentially present 16 bit addresses to the comparing circuits. Two 8 bit input/output ports were dedicated to this function. The kit memory was used to hold the table of addresses to be monitored.

Address information was taken from the microprocessor under test by making connections to its data buss and memory control lines (Figure 2). The 16 address lines were made available to the comparing circuits by using a static memory interface chip as a de-multiplexer, because the microprocessor uses the 8 line data buss both for addressing memory and for transferring data.

Figure 1. Block Diagram of Microprocessor Monitor

The kit was also required to maintain time information. Three central processor registers and the Program Storage Unit (PSU) interrupt line were dedicated to this function.

The 2 MHz kit oscillator was used as a time base. The 2 MHz signal was divided by 2000 using a down counter chip to produce 0.001 second time increment pulses directed to the PSU interrupt line.

The down counter is enabled by a signal obtained from the unit under test so that time increment pulses begin when the unit being monitored is powered up.

CONNECTIONS ARE IDENTIFIED ALPHABETICALLY

THE L CONNECTION FROM PIN 37 GOES TO THE DOWN COUNTER

ALL OTHER CONNECTIONS GO TO STATIC MEMORY INTERFACE PINS IDENTIFIED WITH MATCHING LETTERS

Figure 2. Electrical Connections to CPU and SMI

The kit stores the contents of its three time registers in memory each time the comparison circuits find a match between an address presented by the I/O ports and an address which was called up by the unit under test. When the last monitor address is found, the kit outputs the information it has stored to a teletype printer. If one of the monitor addresses is not found, the time stored for this address and all subsequent addresses will be zero. One I/O port was dedicated to reading and resetting the latched output of the comparing circuits.

PINS WITH X IDENTIFIERS ARE CONNECTED AS SHOWN ON ADDRESS
COMPARISON CIRCUITS SCHEMATIC

RESISTORS ARE 4.7 k Ω

Figure 2 Continued. Electrical Connections to CPU and SMI

System Development

Peripheral hardware items were designed and assembled. The output ports of the microprocessor present data as the complement of the hexadecimal code, while memory address information from the SMI is not complemented. Therefore, the comparing circuits are designed to "exclusive OR" each of the 16 address lines with the corresponding output port line (I/O 1 for upper address lines and I/O 0 for lower address lines). The 16 exclusive OR outputs are connected to AND gates so that a true output will occur only

if all 16 exclusive OR tests are true. One additional AND input was added to assure that a true output would occur only if the CPU Read line was also high. Thus, 17 conditions are required to be satisfied before an address comparison is found to be true. The output of the AND gates is directed to a resetable latch. A true condition, once found, will remain latched until this condition is recognized and subsequently reset (Figure 3).

Then, the remaining hardware was added. The 40 pin clip connector was wired to provide connections (Figure 2) to the static memory interface chip. Address lines, I/O port lines, and time base connections were added.

Light emitting diode hexadecimal display chips along with (inverter) buffers were added to I/O ports 0 and 1. This provides a visual indication of the status of the monitor (Figures 4 and 5). If the monitor fails to find an address during a run, the address not found will remain displayed.

The crystal oscillator circuit on the kit was found to run at 2.003 MHz rather than the intended 2.0 MHz. Crystal load capacitors were increased as shown in Figure 6 and a fine adjust trim capacitor was added. This allowed the crystal operating frequency to be adjusted to agree with other laboratory instrumentation. The circuit shown provides a +25 and -15 Hz adjustment.

Software was prepared in accordance with flow charts shown as Figures 7, 8, and 9. The memory storage space from 0300 to 03E0 is cleared each time the program is started so that data from a previous run is not present. Data counter 0 is set to point to memory location 0300 where data storage will begin while data counter 1 is set to point to memory location 0260 where the table of addresses to be monitored begins. Then, the SMI and PSU unit internal timers are stopped, since they will not be used. The PSU external interrupts are then enabled to allow 0.001 s time increments to be recognized.

The remaining segment of the main program, beginning with Get Address High, is repeated for each address to be monitored. At the end of the address table, FF is loaded into memory. The main program tests for FF when each address is fetched so that it will go to the dump subroutine when FF is found.

After the high and low bytes of a monitor address are outputted to ports 1 and 0, the comparing circuit latch is reset by an output of 04 to port 5.

Port 5 is then read and tested for the presence of 08. The 08 will be found only if the comparing circuit latch is set. The following branch instruction causes port 5 to be read repeatedly until the comparing circuits find an address match.

Figure 3. Address Comparison Circuit Schematic

When a match is found, the address identifier (A1 for the first address and A2 for the second) is stored beginning at address 0300 in RAM (Figure 10). The contents of the time registers, R3 time high, R4 time middle, and R5 time low, are then stored in the next 3 RAM locations. Time increment interrupts are disabled during time storage and enabled afterward. This assures time registers are not altered during execution of store instructions. All interrupts are serviced, however, so that no time increments are lost.

The last instruction jump causes the program to be repeated beginning with Get Address High.

Figure 4. Microprocessor Monitor Connected to Electronic Programmer Printed Wiring Assembly

Figure 5. Static Memory Interface, Down Counter, and Comparison Circuits

C1, 47 pF WAS 10 pF
 C2, 60 pF WAS 10 pF
 C3, 0 TO 8 pF ADDED CAPACITOR

Figure 6. Clock Reference
 Frequency Circuit
 Modification

The PSU Interrupt Service Routine is executed when a time increment pulse is received. Register 5 is incremented decimally with each pulse and registers 4 and 3 are incremented decimally when appropriate carry conditions are found. The number of 0.001 second time pulses received is always equal to the content of registers 3, 4, and 5 until the count of 99 99 99 is exceeded. At that time, register 3 (time high) would have a carry condition. The instruction to go to subroutine dump would then be exercised.

In expected use, this would not occur because times monitored would be less than the 999.999 second capacity of these registers. Normally the Main program is reentered when this subroutine is completed at the memory location which would have been fetched next if the interrupt had not occurred. The program listing is provided in the Appendix.

The main program requires 116 μ s to complete a cycle of storing time information, presenting the next monitor address, reading the comparison latch, and starting again to store time. The interrupt routine requires 29 μ s with no carry, 48 μ s with one carry and 66 μ s with two carry conditions. Assuming a long interrupt may occur between addresses monitored, this program can monitor addresses spaced as close in time as 200 μ s. This allows an 18 μ s margin for leaving and returning to the main program.

Figure 7. Flow Chart of Main Program

PSU INTERRUPT SERVICE ROUTINE

Figure 8. PSU Interrupt Service Routine

ACCOMPLISHMENTS

A laboratory capability has been developed to perform diagnostic work on operating microprocessor based products such as an electronic programmer. Data taken from repeated runs of one programmer are consistent within ± 0.001 second, which provides sufficient accuracy to correlate diagnostic data with production test results and to verify that the product clock oscillator is running within its frequency tolerance limits.

The tables of addresses selected for monitoring can be easily changed so that repeated monitor runs can include a significant sample of the total instructions set used by the product. The printed record of each diagnostic run is provided in an easy-to-read format which can be retained for future reference.

An example of use was demonstrated when an analysis was made of a programmer with modified instructions in its PSU. The instruction difference was found to be a delay routing at memory location 0398 which previously began at memory location 034A.

Figure 9. Subroutine Dump

Other laboratory equipment can be readily used with this system to obtain further information about specific microprocessor functions. An oscilloscope was connected to the set line (from the final AND gate) in the address comparison circuit. This allowed direct viewing of time separation between addresses with additional resolution. All objectives initially planned were achieved.

0007 = 000.000 S	0010 = 007.002 S	M0260-02C0
0107 = 000.000 S	0018 = 010.002 S	M0260=00 07 01 07 01 17 01 22
0117 = 000.002 S	00F0 = 011.503 S	M0268=01 28 01 28 01 37 01 66
0122 = 000.002 S	00FE = 011.518 S	M0270=01 74 01 79 01 80 01 85
0128 = 000.003 S	0101 = 015.496 S	M0278=01 80 01 85 01 80 01 85
012B = 000.115 S	?	M0280=01 80 01 9E 01 A6 01 9E
0137 = 000.231 S		M0288=01 A6 01 87 01 BF 01 C4
0166 = 000.232 S		M0290=01 BF 01 C4 01 DF 01 EA
0174 = 000.255 S		M0298=01 EF 02 0A 03 A4 02 1B
0179 = 000.256 S		M02A0=02 13 02 1B 02 28 02 3B
0180 = 000.259 S		M02A8=02 49 02 85 02 AA 03 05
0185 = 000.262 S		M02B0=03 64 00 5B 00 67 00 B9
0180 = 000.263 S		M02B8=00 D2 00 78 00 10 00 1B
0185 = 000.264 S		M02C0=00 F0 00 FE 01 01 FF FF
0180 = 000.265 S		?
0185 = 000.267 S		
0180 = 000.268 S		
019E = 000.269 S		
01A6 = 000.270 S		
019E = 000.271 S		
01A6 = 000.273 S		
01B7 = 000.274 S		
01BF = 000.277 S		
01C4 = 000.278 S		
01BF = 000.279 S		
01C4 = 000.280 S		
01DF = 000.299 S		
01EA = 000.301 S		
01EF = 000.302 S		
020A = 000.314 S		
03A4 = 000.314 S		
021B = 000.315 S		
0213 = 000.316 S		
021B = 000.317 S		
0228 = 000.318 S		
023B = 000.319 S		
0249 = 000.319 S		
0285 = 000.321 S		
02AA = 000.322 S		
0305 = 000.323 S		
0364 = 000.324 S		
0058 = 003.000 S		
0067 = 005.801 S		
00B9 = 005.801 S		
0002 = 006.051 S		
0078 = 006.301 S		

**TEST
PRINTOUT**

M0300-03E0	
M0300=A1 00 00 00 00 A2 00 00 00	00
M0308=A3 00 00 02 A4 00 00 00 02	00
M0310=A5 00 00 03 A6 00 01 15	00
M0318=A7 00 02 31 A8 00 02 32	00
M0320=A9 00 02 55 AA 00 02 56	00
M0328=AB 00 02 59 AC 00 02 62	00
M033Q=AD 00 02 63 AE 00 02 64	00
M0338=AF 00 02 65 80 00 02 67	00
M0340=B1 00 02 68 B2 00 02 69	00
M0348=B3 00 02 70 B4 00 02 71	00
M0350=B5 00 02 73 B6 00 02 74	00
M0358=B7 00 02 77 B8 00 02 78	00
M0360=B9 00 02 79 BA 00 02 80	00
M0368=BB 00 02 99 BC 00 03 01	00
M0370=BD 00 03 02 BE 00 03 14	00
M0378=BF 00 03 14 C0 00 03 15	00
M0380=C1 00 03 16 C2 00 03 17	00
M0388=C3 00 03 18 C4 00 03 19	00
M0390=C5 00 03 19 C6 00 03 21	00
M0398=C7 00 03 22 C8 00 03 23	00
M03A0=C9 00 03 24 CA 00 30 00	00
M03A8=CB 00 58 01 CC 00 58 01	00
M03B0=CD 00 60 51 CE 00 63 01	00
M03B8=CF 00 70 02 D0 01 00 02	00
M03C0=D1 01 15 03 D2 01 15 18	00
M03C8=D3 01 54 96 00 00 00 00	00
M03D0=00 00 00 00 00 00 00 00	00
M03D8=00 00 00 00 00 00 00 00	00
M03E0=00 00 00 00 00 00 00 00	00
?	

TEST DATA

Continued

Figure 10. Test Printout and Stored Information.

FUTURE WORK

The microprocessor monitor is a versatile laboratory tool that can be modified to optimize specific capabilities. More storage space to accomodate a larger table of addresses could be made available by relocating some of the program or by increasing the data storage capacity of the kit.

If time measurements greater than 9.999 second are not required, the Interrupt Service Routine could be shortened thus speeding up the overall program.

Further work could provide the capability of monitoring and storing addresses fetched at preselected times of interest rather than monitoring times when preselected addresses are fetched.

With minor changes, such as repackaging, the microprocessor monitor could be developed into a printed wiring assembly tester for microprocessor based product. This could test the circuit early in the assembly process when defects are most easily corrected.

Appendix
PROGRAM LISTING

FORMULATOR ASSEMBLER (PEV 3.0)

EPDS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT
				0001	*
				0002	*
				0003	* EXTERNAL INTERRUPT PROGRAM
				0004	*
				0005	* THIS PROGRAM WILL OUTPUT AN ADDRESS
				0006	*FROM THE ADDRESS TABLE BEGINNING AT
				0007	*H'0260'. THE TIME OF DECODE WILL
				0008	*BE PLACED IN MEMORY BEGINNING AT
				0009	*H'0300'.
				0010	*
				0011	ORG H'0000'
				0012	*
0000	290100	0100		0013	JMP MAIN JMP TO MAIN PGM
				0014	*
				0015	ORG H'0100'
				0016	*
:0300000029010003					
	0100	70		0017	MAIN CLR CLEAR REGS. 0-7
	0101	50		0018	LR 0-A
	0102	51		0019	LR 1-A
	0103	52		0020	LR 2-A
	0104	53		0021	LR 3-A
	0105	54		0022	LR 4-A
	0106	55		0023	LR 5-A
	0107	56		0024	LR 6-A
	0108	57		0025	LR 7-A
	0109	80		0026	OUTS 0 CLR OUTPUT PORTS 0
	010A	81		0027	OUTS 1 AND 1
	010B	20A1		0028	LI H'A1' LOAD A1 IDENTIFIER
	010D	58		0029	LR 8-A INTO R8
	010E	2A0300 0300		0030	DCI H'0300' SET DCO=TIME TABLE
	0111	70		0031	CLRRAM CLR CLEAR RAM FROM
	0112	17		0032	ST 0300 TO
	0113	0E		0033	LR 0-DC 03E0
	0114	03		0034	LR A-QL COMPARE DCD TO
	0115	23E0		0035	XI H'E0' H'E0'
	0117	94F9 0111		0036	BNZ CLRRAM BR BACK IF NOT THERE
	0119	2A0300 0300		0037	DCI H'0300' RESET DCO=TIME TABLE
	011C	2C		0038	XDC
	011D	2A0260 0260		0039	DCI H'0260' SET DCI=ADD TABLE
:2101000070505152535455565780B120A1582A030070170E0323E094F92A03002C2A026020FE					
	0120	20FF		0040	LI H'FF' LOAD FF INTO
	0122	87		0041	OUTS 7 PORTS 7 AND F
	0123	8F		0042	OUTS H'F' TO STOP TIMERS
	0124	70		0043	CLR CLEAR PORT E TO
	0125	8E		0044	OUTS H'E' DISABLE M1 INTERRUPTS
	0126	71		0045	LIS H'1' PUT 01 INTO PORT 6 TO
	0127	86		0046	OUTS 6 ENABLE PSU EXT INT
	0128	78		0047	LIS H'8' SET P2=08 FOR TESTS
	0129	52		0048	LR 2-A
	012A	20FF		0049	LI H'FF' SET R7=FF FOR TESTS
	012C	57		0050	LR 7-A
	012D	1B		0051	EI ENABLE INTERRUPTS
	012E	16		0052	LM GET ADD H1
	012F	81		0053	OUTS 1 OUTPUT IT TO PORT 1
	0130	E7		0054	XS 7 IS ADD H1 = FF?
	0131	9404 0136		0055	BNZ OUTLO BR IF NOT TO OUTLO
	0133	29000F 000F		0056	JMP DUMP JMP TO DUMP IF DONE
	0136	16		0057	LM GET ADD L0
	0137	80		0058	OUTS 0 OUTPUT IT TO PORT 0
	0138	74		0059	LIS 4 OUTPUT 4 TO
	0139	85		0060	OUTS 5 RESET DECODE LATCH
	013A	2C		0061	XDC GET DC READY TO STORE

FORMULATOR ASSEMBLER (REV 3.0)

ERPS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT
	A13B	70		0062	NOMACH CLR OUTPUT 0 TO PORT 5
	013C	B5		0063	OUTS 5
	013D	A5		0064	INS 5 READ PORT 5 FOR MATCH
	013E	E2		0065	%S 2 IS PORT 5 = 00?
	013F	94FB	013B	0066	BNZ NOMACH BP IF NOT TO NOMACH
	0141	40		0067	STORE LR A.8 GET IDENTIFIER
:	21012100FFB7BF70BE71B6785220FF571B16B1E7940429000F168074B52C70B5A5E294FB467C				
	0142	17		0068	ST STORE IT
	0143	1F		0069	INC INC ACC
	0144	58		0070	LR 8.A STORE BACK IN R8
	0145	1A		0071	DI D. 1. WHILE STOPING
	0146	43		0072	LR A.3 SET R3 = TIME M1
	0147	17		0073	ST STORE TIME M1
	0148	44		0074	LR A.4 SET R4 = TIME MID
	0149	17		0075	ST STORE TIME MID
	014A	45		0076	LR A.5 GET R5 = TIME L0
	014B	17		0077	ST STORE TIME L0
	014C	1B		0078	EI ENABLE INTERRUPTS
	014D	2C		0079	XDC PNT DC TO ADD TBL
	014E	29012E	012E	0080	JMP NEXT OUTPUT NEXT ADDRESS
				0081	*
				0082	*
				0083	* INTERRUPT SERVICE ROUTINE
				0084	*
				0085	* UPON RECEIVING AN INTERRUPT, THIS
				0086	*SUBROUTINE WILL UPDATE THE CLOCK REGIS-
				0087	*TERS R3,R4,AND R5.
				0088	*
				0089	* ORG H'00A0'
				0090	*
:	0F014200171F581A4317441745171B2C29012E56				
	00A0	50		0091	TINUP LR 0.A SAVE ACC IN R0
	00A1	71		0092	LIS 1 GET 01 INTO THE ACC
	00A2	2466		0093	AI H'66' ADD TO IT '66'
	00A4	D5		0094	ASD 5 ADD DEC1 TO R5=TIML0
	00A5	55		0095	LR 5.A STORE IN R5
	00A6	9212	00B9	0096	BNC RETURN RETURN IF NO CARRY
	00A8	71		0097	LIS 1 GET 01 INTO ACC
	00A9	2466		0098	AI H'66' ADD IT TO '66'
	00AB	D4		0099	ASD 4 ADD DEC1 TO R4=TIMMID
	00AC	54		0100	LR 4.A STORE IT IN R4
	00AD	920B	00B9	0101	BNC RETURN RETURN IF NO CARRY
	00AF	71		0102	LIS 1 GET 01 INTO ACC
	00B0	2466		0103	AI H'66' ADD TO IT '66'
	00B2	D3		0104	ASD 3 ADD DEC1 TO R3=TIMH1
	00B3	53		0105	LR 3.A STORE IT IN R3
	00B4	9204	00B9	0106	BNC RETURN RETURN IF NO CARRY
	00B6	29000F	000F	0107	DUMP TIME REG.'S FULL-DUMP
	00B9	40		0108	RETURN LR A.0 RESTORE ACC
	00BA	1B		0109	EI RE-ENABLE INTERRUPTS
	00BB	1C		0110	POP RETURN TO MAIN
				0111	*
				0112	*
				0113	* DUMP SUBROUTINE
				0114	*
				0115	* THIS SUBROUTINE WILL OUTPUT AN ADDRESS
				0116	*FROM THE ADDRESS TABLE AND PRINT THE TIME
				0117	*THE ADDRESS WAS FOUND IN THE FOLLOWING
				0118	*FORMAT...
				0119	* 0007 = 000.002 S
				0120	*AN H'FF' IN THE ADDRESS TABLE SIGNALS
				0121	*THE END OF THE ADDRESS TABLE AND THE
				0122	*THE END OF THE SUBROUTINE. IT THEN

FORMULATOR ASSEMBLER (REV 3.0)

EPDS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT
				0123	*JUMPS TO UTILITY ROM AT H'8080'.
				0124	*
	93E5	0125		TTYO	ORG H'83E5'
	83D6	0126		TTCR	ORG H'83D6'
	90EC	0127		FDP1	ORG H'80EC'
				0128	*
				0129	ORG H'000F'
				0130	*
:1C00A00050712466D5559212712466D454920B712466D353920429000F401B1C05					
	000F	1A		0131	DUMP DI 0.1. FDP NOW
	0010	2A0260	0260	0132	DCI H'0260' PNT DC0 TO ADD TBL
	0013	2C		0133	XDC
	0014	2A0300	0300	0134	DCI H'0300' PNT DC1 TO TIME TBL
	0017	2883D6	83D6	0135	PI TTYO OUTPUT CR,LF
	001A	2883D6	83D6	0136	REENTR PI TTYO OUTPUT CR,LF
	001D	2800D0	00D0	0137	PI DELAY DLY FOR SLOW TTY'S
	0020	2C		0138	XDC GET DC0 FOR ADD'S
	0021	16		0139	LN GET ADDRESS
	0022	23FF		0140	XI H'FF' IS ADD=FF ?
	0024	9404	0029	0141	BNZ CONT BNP IF NOT TO CONT
	0026	298080	8080	0142	JMP H'8080' JMP TO UTILITY IF DONE
	0029	20FF		0143	CONT LI H'FF' ADD H'FF' TO DC TO
	002B	8E		0144	ADC DECREMENT I
	002C	2880EC	80EC	0145	PI FDP1 OUTPUT ADDHI
:21000F001A2A02602C2A03002883D62883D62800D02C1623FF940429808020FF8E2880EC2849					
	002F	2880EC	80EC	0146	PI FDP1 OUTPUT ADDLO
	0032	2020		0147	LI H'20' ASCII '20' = 'SPACE'
	0034	51		0148	LR 1,A PUT IN R1
	0035	2883E5	83E5	0149	PI TTYO OUTPUT SPACE
	0038	203D		0150	LI H'3D' ASCII '3D' IS '=' SIGN
	003A	51		0151	LR 1,A PUT IT IN R1
	003B	2883E5	83E5	0152	PI TTYO OUTPUT '=' SIGN
	003E	2020		0153	LI H'20' ASCII '20' = SPACE
	0040	51		0154	LR 1,A PUT IT IN R1
	0041	2883E5	83E5	0155	PI TTYO OUTPUT SPACE
	0044	2C		0156	XDC PNT DC AT TIME TBL
	0045	2001		0157	LI H'01' ADD 01 TO DC TO
	0047	2C		0158	XDC PNT DC AT TIME TBL
	0048	2001		0159	LI H'01' ADD 01 TO DC TO
	004A	8E		0160	ADC INCREMENT IT
	004B	2880EC	80EC	0161	PI FDP1 OUTPUT TIME HI
	004E	16		0162	LN GET TIME MIDD
	004F	54		0163	LR 4,A STORE IT IN R4
:2100300080EC2020512883E5203D512883E52020512883E52C20012C20018E2880EC1654207D					
	0050	20FF		0164	LI H'FF' DECREMENT
	0052	8E		0165	ADC DATA COUNTER
	0053	44		0166	LR A,4 GET TIME MIDD
	0054	14		0167	SR 4 PUT MID UP TO MID LO
	0055	17		0168	ST STR IN MEM FOR OUTPUT
	0056	20F0		0169	LI H'F0' SET LOW 4 BIT FLAG
	0058	58		0170	LR 8,A IN R8
	0059	20FF		0171	LI H'FF' DECREMENT
	005B	8E		0172	ADC DATA COUNTER
	005C	2880EC	80EC	0173	PI FDP1 OUTPUT MIDUP
	005F	20FF		0174	LI H'FF' DECREMENT
	0061	8E		0175	ADC DATA COUNTER
	0062	44		0176	LR A,4 GET TIME MIDD
	0063	17		0177	ST STR IN MEM FOR OUTPUT
	0064	20FF		0178	LI H'FF' DECREMENT
	0066	8E		0179	ADC DATA COUNTER
	0067	202E		0180	LI H'2E' ASCII '2E' = 'PERIOD'
	0069	51		0181	LR 1,A PUT IN R1
	006A	2883E5	83E5	0182	PI TTYO OUTPUT 'PERIOD'
	006D	20F0		0183	LI H'F0' SET LD 4 BIT FLAG

FORMULATOR ASSEMBLER (REV 3.0)

EPRS	LOC	OBJECT	ADDR	LINE	SOURCE STATEMENT
006F	58		0184	LR	S.A INTO R8
:21005100FF8E44141720F05820FF8E2880EC20FF8E441720FF8E202E512883E520F0582880F5					
0070	2880EC	80EC	0185	PI	FOPI OUTPUT TIME MIDLO
0073	2000		0186	LI	H'00' SET 8 BIT FLAG
0075	58		0187	LR	S.A INTO R8
0076	2880EC	80EC	0188	PI	FOPI OUTPUT TIME LO
0079	2020		0189	LI	H'20' ASCII '20' = 'SPACE'
007B	51		0190	LR	L.A PUT IT IN R1
007C	2883E5	83E5	0191	PI	TTYD OUTPUT 'SPACE'
007F	2053		0192	LI	H'53' ASCII '53' = 'S'
0081	51		0193	LR	L.A PUT IT IN R1
0082	2883E5	83E5	0194	PI	TTYD OUTPUT 'S'
0085	29001A	001A	0195	JMP	PEENTR REPEAT FOR NEXT ADD
			0196	*	
			0197	*	
			0198	*	DELAY SUBROUTINE
			0199	*	
			0200	*	SHORT DELAY ROUTINE FOR SLOW TTY'S.
			0201	*	DELAY FACTOR=00 FOR MIN. =01 FOR MAX.
			0202	*	
			0203		OR6 H'00D0'
:16007200EC2000582880EC2020512883E52053512883E529001AC8					
00D0	2001		0204	DELAY LI	H'01' LOAD DELAY FACTOR IN
00D2	57		0205	LR	R7 <01=MIN. 00=MAX
00D3	70		0206	LOOP2 CLR	CLEAR
00D4	58		0207	LR	S.A R8
00D5	38		0208	LOOP1 DS	S DECREMENT R8
00D6	48		0209	LR	R.A SET R8
00D7	94FD	00D5	0210	BNZ	LOOP1 IS R8=0? BR IF NOT
00D9	37		0211	DS	7 DECREMENT R7.
00DA	47		0212	LR	R.A SET R7
00DB	94F7	00D3	0213	BNZ	LOOP2 IS R7=0? BR IF NOT
00DD	1C		0214	POP	DELAY DONE-RETURN
			0215	*	
			0216	*	
			0217		END
:0E00D00020015770583B4894FD374794F71C8C					
:12000001000101013401014F0100B701001E0100860106					
:0000DE0022					

FORMULATOR ASSEMBLER (REV 3.0)

SYMBOL	TYP	VAL	REFERENCES
CLRRAM	L	0111	0036
CONT	L	0029	0141
DELAY	L	00D0	0137
DUMP	L	000F	0107 0056
FOPI	R	80EC	0188 0185 0173 0161 0146 0145
LOOP1	L	00D5	0210
LOOP2	L	00D3	0213
MAIN	L	0100	0013
NEXT	L	012E	0080
NOMACH	L	013B	0066
OUTHI	L	012F	
OUTLO	L	0136	0055
PEENTR	L	001A	0195
RETURN	L	00B9	0106 0101 0096
STORE	L	0141	
TINUP	L	00A0	
TTCR	R	83D6	0136 0135
TTYD	R	83E5	0194 0191 0182 0155 0152 0149

END ASM
00 ERRS
00 EPRS
>

DISTRIBUTION

	Copy
R. Bulcock, DOE-KCAO	1
V. C. Vespe, DOE-ALO	2
J. A. Freed, LASL	3
S. J. Buginas, LLL	4
H. M. Brinkmeier, Monsanto	5
J. Baremore, SLA	6
C. R. Blaine, SLA	7
J. Carruthers, SLA	8
A. Cooper, SLA	9
C. A. Harris, SLA	10
E. McKelvey, SLL	11
D. Tanner, SLL	12
D. H. Bahm, D/131, FC39	13
W. E. Simes, D/131, FC39	14
J. E. Long, D/531, 1A46	15
H. T. Barnes, D/554, BD50	16-17
L. Stratton, D/554, 2C44	18-20
C. R. Beary, D/755, 1A42	21
R. P. Frohberg, D/800, 2A39	22
F. A. Spies, D/800, 2A39	23
P. F. Arnold, D/845, MF39	24
R. Biase, D/845, MF39	25
R. C. Douglass, D/845, MF39	26
R. C. Harnden, D/845, MF39	27
J. J. Hoefer, D/845, MF39	28
R. G. Morgan, D/845, MF39	29
R. J. Powell, D/845, MF39	30
E. J. Rolfe, D/845, MF39	31
L. W. Siever, D/845, MF39	32
D. J. Stevens, D/845, MF39	33-34
D. A. Cummings, D/863, FB39	35
A. V. Grimm, D/863, FB39	36
J. H. Swafford, D/863, FB39	37
D. L. Willyard, D/864, MF39	38
R. E. Kessler, D/865, 2C40	39

BDX-613-2072 (Rev.)

MICROPROCESSOR CIRCUIT MONITOR, D. J. Stevens,
Final, February 1979.

A microprocessor based diagnostic system was developed to monitor a microprocessor circuit while it is operating. The system determines if selected memory locations are addressed in the correct sequence and records time to the nearest 0.001 second when each selected memory location is addressed.

ELECTRICAL: Microprocessor Monitor

MICROPROCESSOR CIRCUIT MONITOR, D. J. Stevens,
Final, BDX-613-2072 (Rev.), February 1979.

A microprocessor based diagnostic system was developed to monitor a microprocessor circuit while it is operating. The system determines if selected memory locations are addressed in the correct sequence and records time to the nearest 0.001 second when each selected memory location is addressed.

MICROPROCESSOR CIRCUIT MONITOR, D. J. Stevens,
Final, BDX-613-2072 (Rev.), February 1979

A microprocessor based diagnostic system was developed to monitor a microprocessor circuit while it is operating. The system determines if selected memory locations are addressed in the correct sequence and records time to the nearest 0.001 second when each selected memory location is addressed.