bcc	title	prefix/class-number.revision
	APU Diagnostic Program	APUD/W-47

checked	authors	approval date	revision date
checked	Butler W. Lampson R. R. Van Tuyl	8/17/70	
approved		classification: Working Paper	
		distribution: Company Private	pages: 5

ABSTRACT and CONTENTS

The comprehensive diagnostic for all non-branching APU instructions is described. All information necessary for operating the program is (hopefully) given.

APU DIAGNOSTIC

This program tests all ITP instructions and features except

branches
UPS, UPOT, UPIN, XCA, CCA, EXU
pops
addressing (but the program uses all addressing features except absolute indirection)
traps (illegal opcodes and memory protection)
input/output

It works in the following way.

Instructions are described by a table called INSTAB. Each entry in this table describes one instruction and contains 8 words:

Ø) the instruction under test. If it addresses memory, its address field contains NOP2.

1) the address of the result table for the instruction
2) the address of a word containing the value which should be in A after execution of the instruction
3) same for B
4) same for X
5) same for NOP2
6) same for skip. The value is zero if the instruction does not skip, -1 if it does.
7) the number of words in a result table entry for this instruction.

The contents of A,B,X and NOP2 before instruction execution are contained in SA, SB, SX, and OP2; the addresses of these locations are therefore frequent occupants of words 2-5 of an instruction table entry. For example, the

instruction table entry for STA is

```
STA   NOP2
Ø
SA
SB
SX
SA
=Ø
Ø
```

indicating that STA has no result table, does not change ABX, never skips and leaves the old contents of A in the location addressed by it.

The result table takes care of the possibility that one of the results of an instruction might not be the same as any of its inputs. In this case, the result table contains an entry for each set of arguments which may be presented to the function. The address of the entry is put into X before the instruction table is accessed, so that an indexed address in words 2-6 will refer to the result table entry. The table is set up by running the diagnostic with the SKE instruction which does the comparison replaced by an STA. Here is the instruction table entry for SKR

```
SKR    NOP2
XXXX
SA
SB
SX
2B7
2B7+1
2
```

It has two result table words, one for the contents of the effective address and one for skip/ noskip. The address of the current result table entry is initialized from word 1 of the instruction table and is incremented by word 7 after each execution of the instruction under test. Its current value is kept in RESPTR.

Arguments are presented to instructions (the same arguments for all instructions) from a table called ARGTAB. The four words of this table addressed by ARGPTR are used to set up A, B, X and NOP2 before the instruction under test is executed. ARGPTR is initialized to ARGTAB when testing of an instruction is started and is incremented by one after each trial. NRES trials are made for each instruction. Note that each word of ARGTAB is used successively as A, B, X and NOP2 (except the first and last 3).

The following flags and pointers control execution of the diagnostic. Numbers in parentheses are the values they are initialized to

BEGINS (INSTAB) address of first INSTAB entry to test
ENDINS (EINST-8) address of last INSTAB entry to test
BEGCHM (Ø) address of first memory location included in checksum
ENDCHM (10000B) address of last memory location included in checksum
EWAITF (-1) if -1, wait for carriage return to be typed after an error before proceeding, if Ø, proceed at once.
DWAITF (-1) if -1, wait for carriage return to be typed after testing all instructions before looping to test them again; if Ø, proceed at one.

IWAITF (Ø) if -1, wait after testing each instruction before looping to test the next one; if Ø, proceed at once.

TRPTF (Ø) if -1, repeat the current test with the current operands indefinitely; if Ø, sequence through tests normally.

IRPTF (Ø) if -1, repeat the test of the current instruction indefinitely; if Ø, sequence through instructions normally.

CHKSF (-1) if -1, memory is checksummed after each instruction has been tested to see if execution of that instruction (or foulups in the diagnostic) improperly altered memory.

The following locations contain useful information:

INSPTR	address of INSTAB entry for instruction under test
RESPTR	address of result table entry for operands under test
ARGPTR	address of 4 words used to initialize ABX and NOP2
LCTR	number of complete loops performed
ICTR	number of instructions tested in current loop
TCTR	number of trials of current instruction

The following locations contain entry points to the various loops:

ESTART	start program
EMTEST	test all instructions
EMTLOOP	test one instruction
EMPSTLP	test one case of one instruction

There are two kinds of errors: comparison and checksum. They both increment counters and dump information into

a ring buffer between the address contained in cells LCOMP and COMPP. A pointer to both the Message Array (MSGA) and LCOMP, COMPP is contained in the first part of the program.

The Message Array contains:

Index	Contents
Ø	Counter of number of loop of the main test
1	Compare error
2	Checksum error

The program loads cells Ø - 14ØØØB. The program begins at 624ØB. In the event that the program is requested to wait, cell 6ØØ1B is set to Ø and the program waits until it becomes non-zero.