60015

October 21, 1965

DISTRIBUTION SHEET

DW Burbeck

HP Lee

E Vance

P Jennings

RA Decker

G Futas

H Mathews

D Dahm

R McGee 645 Software

H Frick

JL Gildersleeve

SG Gangwere

RD Greenway

October 21, 1965

ON-LINE TESTING IN A 636 TIME-SHARING SYSTEM

SUMMARY

The following pages contain a first pass discussion of on-line testing in a 636 System. This is not a statement of what is going to be in the System, but it is what we feel should be a part of a large time-sharing multi-processor system in order to keep it on the air for a greater part of the day than is possible with off-line T&D. H. Mathews' write-up of September 22, 1965 -- Peripheral T&D Interface with 645 Software Monitor -- could be inserted at section 3.2. of this document.

At this early date in the software and applications development, it is probable that necessary areas have been overlooked, so I would appreciate any comments or suggestions that you may have.

K.L. Mikus

ON-LINE TESTING IN A 636 TIME-SHARING SYSTEM

1.0 DEFINITION AND PURPOSE

On-line tests are defined as programs that may be run to functionally test a unit or subsystem during the normal operation of a time-sharing system. The part of the system being tested will be de-allocated from the equipment configuration available to software or the users for the duration of the test. It should be possible for the Operating Supervisor (OS) to automatically call in the tests during idle time, for scheduled Preventative Maintenance, Confidence Testing, or upon discovering unrecoverable error conditions. Also any test could be called in by the Product Service Engineer via the Maintenance Console. By using the technique of on-line testing, preventative maintenance and failure testing may be done on non-unique units without requiring that the whole system be involved. Where there are many like units, the throughput of the system should not even be decreased.

1.1 Assumptions

1.1.1 Hardware System Configuration

The function of on-line testing is not particularly dependent on a specific equipment configuration. How vital the units are to the system (i.e. the system is highly dependent on an interlaced memory module or the Fire Hose Drum while it is not on one of several card readers or tape units) determines the effectiveness of on-line testing. However, even if the drum or a memory module were down the system could struggle along at a highly reduced rate.

The key feature of the hardware is the ability to allow easy manual reconfiguration by unit switching from a central point rather than unplugging of cables and resetting rows of switches. A specific reconfiguration that should be possible is to split a dual system into two similar systems. In addition, the tests must be able to identify which physical unit they are testing -- including which processor.

Any features in the hardware that could give more information than present hardware does about its state at any given time or provide improved error detection, would enhance the value of on-line testing--as well as enable eventual off-line automatic diagnosis.

1.1.2 Operating Supervisor

Certain privileges and privileged information must be granted to on-line tests by the Operating Supervisor. Just as hardware reconfiguration is necessary, so is de-allocation of any unit, including a processor in a multi-processor system, from the software and user until the test is complete. In all tests it is mandatory to know which physical unit is being tested.

For peripheral testing (including the Fire Hose Drum and GIOC or IOC) all status and queue information must be made available to the test. For processor tests, all faults are regarded as pertinent information (MME's, too) and should be turned over to the test for processing.

The Timer presents another exception case for the OS. Processor tests should not be interrupted at certain times and so the Timer must be adjusted to prevent this. In peripheral tests, timing is often important, especially to determine if events have not occurred which should, so they must be able to have the Timer set at their required values. There will be problems here if the OS cannot keep an account of elapsed time.

Some processor tests must be run in the Master mode and, even more significantly, others must be run in the Absolute Mode. If these programs are regarded as parts of the OS rather than as user programs, it should be possible to run these kind of tests on-line At worst, one processor and a portion of memory would be deallocated for the Absolute Mode tests.

A desirable feature to be included in the OS is for it to keep a record of errors and call in test programs when some predetermined level is reached.

2.0 TYPES OF ON-LINE TESTING

There are several specific types of testing that are inherently unsuitable to be run on-line. Diagnostics are one of these, as the ability to diagnose implies special modes of operation such as step cycle, partitioned operation of the logic and deliberate error producing sequences, all of which would interfere with normal operation. Systems testing would not be allowed, either,

since it means that all or most of the system is being tested simultaneously. Worst case memory, Fault, Privileged instructions, Random Instruction Sequence tests and any others that jeopardize normal operation by forcing errors would also be reserved for off-line. The types of tests that may be run on-line are discussed below.

2.1 Processor and Memory Tests

For many of these tests, the assumption must be made that certain portions of the hardware are working, i.e. any tests run in the slave mode must assume that address appending is functioning properly. In this category would be floating point, register, address modification, instruction (excluding privileged ones, of course) and illegal slave mode functions. The address appending, associative memory and memory tests must be run, at least partially, in the Absolute Mode and are considered basic tests.

2.2 Peripheral Tests

Any peripheral may be tested on-line as long as the appropriate interface is provided (see Peripheral T&D Interface with 645 Software Monitor; H.D. Mathews, 645 T&D; September 22, 1965.) It is not desirable to treat the Fire Hose Drum as a normal peripheral, i.e. if it is to be tested, special provisions must be made, since it is really an auxiliary memory rather than an I/O device.

The diagnostic modes of the GIOC will be avoided also, primarily as extensive testing of the GIOC, unless necessary will reduce the system throughput.

3.0 INTERFACE WITH OPERATING SUPERVISOR

3.1 Processor Interface

3.1.1 Program Header.

W1	2 0 2 0	Phase #	Program Type Code	2 0
W2	Program Identification		Test Number	Sub Test ID
W3	MBZ		# Of Words Of Memory Used	
W4	Timer Value			
*W5	Processor # or Memory Port		Address of SCU Info.	
W6	MBZ		Address of Test Xfer Table	
W7	MBZ		Number of Entries in Transfer Table	
W8	DBR			
W9	SDW			
W10	Not Used			
W11	Not Used			
*W12	Time When Last Run			

*Information provided by Operating Supervisor.

WORD

1. Bits 18-29 Program Type Code
   1. Processor - 40 (Octal)
   2. Memory - 45 (Octal)

   Bits 12-17 Phase Number - Test level, 0-9

2. Bits 0-17 Test identification - three alphabetic BCD characters.

   Bits 18-29 Test Number - two numeric BCD characters.

   Bits 30-35 Subtest Identifier - one alphabetic character.

3. Bits 18-35 Amount of memory used by this test.

4. Bits 0-23 Contains a count for presetting the Timer.

5. Bits 0-3 Processor Number or Memory Port.

   Bits 18-35 Address of SCU or STCD information.

6. Bits 18-35 Address of Test Transfer Table - set up by the test. It will be used with the test number by the OS and Product Service man to call in a test.

7. Bits 24-35 Number of words in the Test Transfer Table.

8. If this word contains all zeros the test is to be started in the Absolute Mode. If non-zero it contains information to be loaded into DBR and the test will be started in the Appending Mode. The Descriptor Segment absolute address base may be relocated.

9. If word 8 is zero this is not used. If word 8 is not zero this word contains the SDW to be used to access the test. The segment absolute address base may be relocated. The pointer to the test segment will be zero (0 in bits 0-17 of an ABRn).

10-11 Not used for processor tests.

12. Time this test last run.

3.1.2 Transfer of Control Between OS and Tests

After the program has been searched for and loaded, control will be passed to the T&D program in the normal manner--either a TRA or a TRA with Bit 29 = 1, depending on Word 8 of the T&D Program Header, to the Test Transfer Table modified by the test number.

A test will be terminated by issuing a MME with the characters "T&D" in the operand (or any other suitable operand). This should cause the OS to interrogate the Maintenance Console to see if the PSE wants to run any more tests, and if not, terminate the T&D run. Before executing this MME, the test program will clear its MME fault location to zeros to signify to the OS to process the fault.

3.1.3 Fault Handling

Test program locations 0-63 (relocatable) are the program's own fault vector where control should be transfered after a fault. Either an SCU or STCD, as appropriate, will be assumed to have been used by the OS in the actual fault vector and this information must be available to the test.

3.1.4 Error Output Messages

The test program could use the Type or Print Routines (depending on the Maintenance Console Switch setting -- see 3.1.6) contained in the OS, in order to save space. The test programs will determine the switch settings and set up the print line (including slew character or carriage returns). If the typewriter is selected, it will be the one on the Maintenance Console.

3.1.5 Maintenance Console

3.1.5.1 The console should consist of an I/O typewriter, thirty-six toggle switches, register displays and other debugging aids

3.5.5.2 Switch settings.
See page 19 of Peripheral T&D Interface with 645 Software Monitor.

3.2 Peripheral Interface

See Peripheral T&D Interface with 645 Software Monitor

4.0 OPERATION

The repertoire of tests should be run on some maximum time basis such as a 24 hour interval. In addition, they could be brought in during idle time -- if there ever is such a thing. In any case, the tests would have low priority in the system until the maximum interval had passed, at which time they would obtain top priority. The

Product Service Engineer or any one else can also call the programs in via the Maintenance Console. This Maintenance Console interface must be in the OS.

Kathy L. Mikus
System Design Engineering

KLM:mhw