bcc

title	System Initialization and Error Recovery	**prefix/class-number.revision** SINIT/W-19
checked	**authors** Larry L. Barnes Larry L Barnes	**approval date** 9/12/69
checked		**revision date**
approved Mel		**classification** Working Paper
		distribution Company Private
		pages 9

ABSTRACT and CONTENTS

This document discusses the procedures for starting the system and recovering file storage after a crash. It covers microprocessor initialization algorithms, but does not cover the procedures for dumping or reloading the disk. Later versions will include formats and detailed recovery algorithms.

1. Definitions of Startup Conditions

The M1 system is unique in requiring a large amount of information to be present in core before the various processors can run. A minimum of 6K is needed before the CPU can operate: the CHT, a context block, and a page of code. Before the AMC can run the drum bit table and the DHT must also be initialized. The micro-scheduler requires the PRT to be initialized before running. The CPU must initialize all CHIO tables before it can operate normally.

The system will be operational when all processors are running in a normal state. Once the M1 becomes operational, normal but privileged user programs can finish the task of getting the system into normal operation. These programs are termed recovery procedures. Upon completion of recovery, the system is said to be in the running, or normal, state. In addition to these two states, we distinguish two starting states: tape start and disk start. Tape start occurs whenever the system must be loaded onto the disk or drum; disk start, when only the core must be reloaded.

2. Hardware Requirements

The hardware required for starting and stopping the system consists of

1) two push-buttons labeled STOP and GO,

2) a CONTINUE flip-flop, and

3) twenty-two toggle switches.

The GO button, CONTINUE flip-flop, and the toggle switches form a twenty-four bit register, (SCR) which will be read by the UTP and stored in memory. The STOP button will produce a pulse approximately 1μs long after the button has been pushed and released. This STOP signal will force all components of the system into an initial state, excepting only the fast memory. All micro-processors will take a forced branch to Ø, the TSUs will be reset, and the Model 30 will be interrupted and the M1 channel reset. There will also be an ITP instruction to set the STOP signal.

The GO button will set a flip-flop when it is released; the flip-flop is part of the 24-bit SCR and the GO flip-flop will be reset after the SCR is read. The CONTINUE flip-flop is set by the GO button and reset by the STOP button, but not the STOP signal.

When the microprocessor P-counter is cleared, each microprocessor performs its initialization sequence. The microscheduler performs the following algorithm.

1) Read the switch register. Go to step 4) if CONTINUE

2) Read the switch register. Go to step 2) if $\overline{\text{GO}}$.

3) Store the switches in SRMEM. Go to step 5).

4) Read SRMEM. Go to step 2) if $\overline{\text{CONTINUE}}$ (in SRMEM).

5) Send strobes to CHIO and AMC. Wait for strobe, then execute normally.

The effect of this algorithm is to set SRMEM from the switches if the STOP button is pushed, otherwise the current contents of SRMEM are used. The second alternative is used for ITP-initiated start sequences.

The initialization sequence for all microprocessors, except the micro-scheduler, is to wait for a request strobe. The micro-scheduler will not send the strobe until it has set up SRMEM properly.

The interpretation of the switch register is only partially defined. The following assignments have been made:

Bit	Meaning
Ø	GO signal
1	CONTINUE signal
2-3	Action on GO or start-up. 00 - Nothing 01 - Tape Start 10 - Disk Start 11 - Restart
4-8	Select one of thirty-two areas to save core in and read the system from.
9-21	Currently unassigned

22	Start the ITP	These two take priority over the others
23	Self-fill the ITP	

The meaning of bits 4 to 8 is discussed in section 4.

When the micro-scheduler is running normally, it will update SRMEM in core and set the light-register from LRMEM. This will be done once every millisecond. If the CPU shuts off the microscheduler it will not update SRMEM.

3. Tape Start Sequence

The tape start procedure is executed by the CHIO to load core with a bootstrap program which proceeds to load the disk or drum with the files required to run the system. After the CHIO receives a request strobe, it reads SRMEM and performs the following algorithm.

1) If switch bits 2-3 are not 01 (tape start), send a restart control character to the model 3Ø and go to step 7.

2) Send a tape-start control character tothe Model 3Ø.

3) Read a character from the Model 3Ø. Go to step 6) if it equals 77B. Otherwise set up to read into the page specified by the character value, and initialize the check sum.

4) Read 8192 six-bit bytes from the Model 3Ø, pack them and store them into the selected page. Compute a check sum by exclusive OR'ing the current word into the sum.

5) Read four more bytes and update the check sum; check it for zero. Send either 'valid data' if the sum is zero or 'invalid data' if non-zero to the '3Ø. Go to step 3).

6) Send a request strobe to the UTP.

7) Wait for a request strobe, then initialize and start normal execution.

This procedure allows us to reliably load arbitrary pages of core from the Model 3Ø.

The bootstrap program will proceed to initialize the tables for all other micro-processors including the CHIO and send them request strobes so that it can proceed to load the disk. The loading procedure consists of:

1) Reading the unique name and disk address from the 36Ø/3Ø via the low-speed line,
2) Zeroing the disk class code for this page,
3) Creating a new page for the information in the normal manner,
4) Reading the information from the Model 3Ø (over the high-speed channel) into the new page, and
5) Reading the check sum over the low-speed line, checking the information.

If the transmission was valid, we send a 'valid data' character to the '3Ø and move the page to the disk. Otherwise we send an 'invalid data' character to the'3Ø and destroy the page.

This procedure is repeated for each page of the disk to be loaded. When a unique name equal to -1 is received in step 1) above, the bootstrap program will initiate a disk start.

4. Disk Start and Restart

The disk start and restart procedures are similar. The AMC will perform the following algorithm after its P-counter is cleared.

1) Wait for a request strobe.
2) Read SRMEM.
3) Go to step 8) if bit 2 is reset (not disk start).
4) Write out the contents of the first 64K of memory on the crash area if restarting (bit 3 of SRMEM on).*
5) Read the system image into the first 64K of memory.*
6) Store the old value of SRMEM in memory to tell the monitor if this is a disk start or restart
7) Send the UTP a request strobe.
8) Wait for a strobe, then initialize and start normal execution.

The system will then proceed with recovering file space. On a restart the system must:

1) Read the old CHT from the crash area and write all dirty pages onto the drum.
2) Reconstruct DHT from the crash area version, then move all dirty drum pages to the disk.

During these steps the system must carefully check the old CHT and DHT for consistency and make a list of pages which might contain invalid information. Then for both disk start and restart the system will scan all MIBs, CBs, and IBs to

* These addresses will be computed from bits 4-8 of SRMEM.

reconstruct the disk bit table. After this step the system is capable of normal operation.

The disk and drum areas to be used by the AMC are computed from the five-bit field found in bits 4-8 of the switch register. Bits 4-5 are used to select the TSU; bit 8, for the drum or disk number. The drum band number is computed by shifting bits 6-7 left 2. The AMC should assume that the drum is 48K around. (We will use the extra 16K on the large drum normally.) The disk band number is the product of bits 7-8 and the number 12 (11ØØ$_2$). The locations of the crash and system areas on the drum are thus:

Bits 7-8	Crash Area*	System Area
Ø	ØØØ-Ø27,Ø4Ø-Ø47	1ØØ-127,14Ø-147
1	2ØØ-227,24Ø-247	3ØØ-327,34Ø-347
2	4ØØ-427,44Ø-447	5ØØ-527,54Ø-547
3	6ØØ-627,64Ø-647	7ØØ-727,74Ø-747

Similarly for the disk:

Ø	ØØØØ-5,ØØ2Ø-5,	Ø14Ø-5,Ø16Ø-5,
	ØØ4Ø-5,ØØ6Ø-5,	Ø2ØØ-5,Ø22Ø-5,
	Ø1ØØ-5,Ø12Ø-1	Ø24Ø-5,Ø26Ø-1
1	Ø3ØØ-5,Ø32Ø-5,	Ø44Ø-5,Ø46Ø-5,
	Ø34Ø-5,Ø36Ø-5,	Ø5ØØ-5,Ø52Ø-5,
	Ø4ØØ-5,Ø42Ø-1	Ø54Ø-5,Ø56Ø-1
2	Ø6ØØ-5,Ø62Ø-5,	Ø74Ø-5,Ø76Ø-5,
	Ø64Ø-5,Ø66Ø-5,	1ØØØ-5,1Ø2Ø-5,
	Ø7ØØ-5,Ø72Ø-1	1Ø4Ø-5,1Ø6Ø-1

* All numbers are octal

3	11ØØ-5,112Ø-5,	124Ø-5,126Ø-5,
	114Ø-5,116Ø-5,	13ØØ-5,132Ø-5,
	12ØØ-5,122Ø-1	134Ø-5,136Ø-1

We believe these sets of addresses, of which two in general are used, are sufficiently dispersed to guard against head failures and bad surface areas.