Index Panel

SLC Control Program

INDEX

Slac's Software Engineering newsletter

February 23, 1989 — All That Fits is News to Print — Vol. 3, No. 8

Configuration Index Display Speedup

February 21, 1989

Author: *Tom Himel* **Subsystem:** *Configurations* **User Impact:** *Small*
Panel Changes: *None* **Documents:** *No* **Help File:** *None*

As we have accumulated more and more configurations, the display of the index has gotten slower and slower. For example, it took five seconds to look at the scratch index of sector 2–30 configurations. This contains configurations 835–1100.

The software has been improved so it no longer wastes time looking for configurations which have been deleted (1–834 in the above example). To display the above index now only takes one to two seconds.

The DISPLY LAST PAGE button now really gets you to the last page instead of the next page.

BPM Scaling Software Changes

February 22, 1989

Author: *Linda Hendrickson* **Subsystem:** *BPM* **User Impact:** *Small*
Panel Changes: *None* **Documents:** *Previous* **Help File:** *Previous*

Several minor changes have been made to BPM scaling software in a recent release. A bug has been fixed which allowed rescaling when fudge factors exceeded the reasonability limits. In addition, the reasonability limits for fudge factors have been changed from (minimum .5, maximum 2.) to (minimum .6, maximum 2.4) to accomodate the fact that most BPMs read low values compared to toroids.

A problem occurred when LI00 was rescaled due to special handling of the attenuation in this sector. The capability has been added in the software to rescale intensities without scaling attenuations if the correct BPMS HSTA bit is set, and the appropriate bit has been set in all LI00 units.

An open issue remains regarding scaling in the ARCs and other multiplexed regions. The current version of the software calculates independant attenuation scale factors for each BPMS unit, and it may be required that the attenuation should be the same for all BPMS units associated with a given BPMP processor.

Klystron Veto System

February 22, 1989

Author: *Tom Himel* **Subsystem:** *Klystrons, BPMs* **User Impact:** *Moderate*
Panel Changes: *Many* **Documents:** *No* **Help File:** *Yes*

Introduction

The purpose of the klystron veto system is to provide a fast signal to all micros that a klystron has missed firing on a beam pulse. This will allow users to ignore the pulse. The main benefit of this is that it will help keep feedback loops from being driven crazy by the few percent of the pulses where a klystron fault has occurred. It can also be used by Mark II to ignore event triggers where a klystron faulted.

An example of the benefit of this is the energy loop for the positron extraction line. Presently, we don't use this loop because it happened too often that a klystron would be faulted during the few seconds when it was reading the BPMs. It would thus measure an energy which was 230 MeV below the normal, and adjust all the bend magnets in the extraction line. When the klystron came back, the energy would be too high, the beam would hit the beam pipe and trip off the accelerator. This happened so often that the loop isn't used and we adjust that energy by hand. The klystron veto system is explicitly intended to cure problems like this.

The veto system has been in the works for a long time and many people have worked on it. The software is brought to you by Tony Gromme, Mick Flores, Jim Hodges, John Zicker, Keith Jobe, Lou Sanchez, and Tom Himel.

Hardware

The hardware consists of a new VETO CAMAC module in each micro. Each one of these has room for 16 veto inputs. In the typical linac micro eight of these are used, one for each klystron. Each input is generated by a PIOP which checks its klystron's performance and signals the veto module via the MKSU if full power wasn't generated. The VETO modules are connected together with two twisted pairs down which a 32-bit serial stream of data is transmitted after each beam pulse. Encoded in that stream is the geographical location of where the veto was generated.

In somewhat more detail, here is what happens. After each beam pulse (at 360 Hz), micro MP00 sends out a 32-bit word of all zeros on one of the twisted pairs known as the write bus. That word gets passed from one veto module to the next, going around the arc to the final focus and then back to LI00. Each veto module just passes the word on unless one of its enabled inputs indicates a veto. In that case, it turns on a bit in the 32-bit word corresponding to the geographical location of the veto module (e.g. CID, LI00, NDR, LI20–30). The veto modules do not record the data at this time since it doesn't contain information from modules later in the daisy chain. They only record whether there was a parity error and the fact that the data arrived. At LI00 the data is taken off the write bus and put into the second twisted pair known as the read bus. The data gets passed back up torwards MP00. Each veto module records the data word as it goes by so that its micro can later read it to find out whether a veto has occurred anywhere in the accelerator.

Basic Software

The PIOP must decide if the klystron fired properly. In most cases, it knows that the klystron didn't fire, because it told it not to in order to protect it. The situations where this happens are: PIOP can't talk to the MKSU or PAD, there are too many bad pulses compared to good pulses, the klystron input water temperature is <10 or >45 degrees centigrade, the klystron delta water temperature is out of tolerance, the klystron or wave guide vacuum is bad, the klystron or wave guide water flow is bad, the klystron electromagnet current is <10 amps, the klystron beam is over current, over voltage, or there is a reflected energy fault. In these cases it turns off the klystron for about one second. These types of problems are signalled by what is known as a *long* veto. Namely, the PIOP not only vetos the beam pulse where the klystron should have fired and didn't, it also vetos the next pulse 1/360 of a second later in order to flag that the veto will

continue for some time. In a later enhancement of the veto system, these long vetos will be used to tell the MPS system to fire the single beam dumpers to protect the arcs and final focus from the off energy beam. Finally, there is the case where the klystron fires but doesn't put out its full energy. When this happens, the next pulse is usually OK. The PIOP generates a *short* veto (of only that one beam pulse) if the output power is less than 80% of normal.

After the PIOPs have generated their veto signals, they are distributed to all the veto modules as described in the previous section. Before reading its BPMs, each micro reads its veto module. If there is a veto coming from a region upstream of the micro, the data is flagged as bad and is not used in the BPM average and a status bit is set to indicate that some data was vetoed. If all the beam pulses used in the average are vetoed or bad for some other reason, the bad status bit is set and then no VAX software will use that BPM data. The data read from the veto module is also saved in a circular buffer, thus recording a history of faults for later display.

To keep defective hardware from locking the veto on, there is a feature in the micro code which automatically disables a veto input if it is solidly on for more than five seconds. An error message is generated when this happens. Note that a bad klystron won't make this happen as the PIOP only generates vetos for pulses where the klystron was supposed to fire and no klystron is supposed to fire at 360 Hz. Hence the only way to have the veto solidly on is to have bad veto hardware (typically the MKSU is at fault or there is an open cable from the MKSU to the veto module).

The BPM displays now show whether the veto system is enabled and indicate whether the data was vetoed. If all pulses were vetoed, the line is red and the word "vetoed" is written in the right hand column of the display. If several pulses are being averaged and only some of them are vetoed, then the x,y coordinates are yellow, and a green vetoed is put in the right hand column. There is a new button (on the BPM calibration panel) which allows one to enable or disable the use of the veto system for the SCP.

No special changes have been made to feedback, correlation plots, or steering. If their BPM data is vetoed, the status will be bad, and the programs already know not to use such data.

Control and Displays

Shown in Fig. 1 is the new VETO CONTROL panel which is reached from the SPECIAL DISPLAY panel. It allows one to turn veto modules online or offline, to change which klystrons are enabled to generate vetos, and to return them to their default value. It also allows the effects of the veto system on BPM readout to be globally enabled or disabled. Using this enables all SCPs and feedback to start or stop using the veto system the next time they start taking BPM data.

Many displays are available. There are histograms of the fault rate (in units of faults per hour) binned by geographical region, or by micro within a geographical region, or by klystron within a micro. An example is shown in Fig. 2. There are also histograms showing the deadtime fraction (in percent) caused by the vetos. These are binned as above. A time history of the faults within a micro is available. All of the above displays use data from the circular buffers kept by the micro. They can all be set to use data from a particular beam code, or all beam codes. They can also be set to display data for a specified number of hours. If the circular buffers don't contain as much data as you request, all the available data is used.

Finally, there is a system status display which tells which veto modules and klystrons are enabled. Micros which have klystrons which should be enabled and that are not are displayed in red. This display lists the micros in the order that they are connected together in the hardware.

Veto Diagnostics

There is a new VETO DIAGNOSTICS panel which can be reached from the VETO CONTROL panel. It is shown in Fig. 3. If it looks like the veto system is broken, this can be used to tell where the problem lies.

The veto diagnostic puts the micros in a special test mode where they record veto data on 32 pulses. The data recording is synchronized by a special YY, similar to the method used for BPM data acquisition. For test type NORMAL, the normal veto signals are recorded and checked for errors. Klystrons which generate vetos are displayed. For test type WALKING 1 the pattern micro (normally MP00) is told to send out a particular pattern of 32-bit words instead of the zeros it normally sends out. It will first send 1, then 2, then 4, then 8, etc. What arrives on the read bus at each micro is recorded and compared to what should be there. Errors are displayed. Normally, there is enough information available to tell where the system is broken.

Finally, there is a diagnostic feature which allows one to force a PIOP to generate a veto. One can then use the veto displays to see if a veto shows up in the circular buffer. This allows one to check whether the connection from a PIOP to the veto module works properly.

Present Status of the System

The hardware and software are all installed but there are still a few minor problems which we are working on. A few non-essential micros are not yet hooked into the veto system. Three klystrons permanently indicate a veto and thus are disabled. The veto diagnostic sometimes reveals parity errors. These don't seem to happen for real data. Sometimes when you boot a micro, all the veto inputs get disabled and you have to reenable them from the VETO CONTROL panel. Short vetos (based on low RF output) are not yet implemented. All of these items are being worked on and we hope to have everything done in a couple of weeks. In the mean time the system should be useful although it will take some care to keep it up.

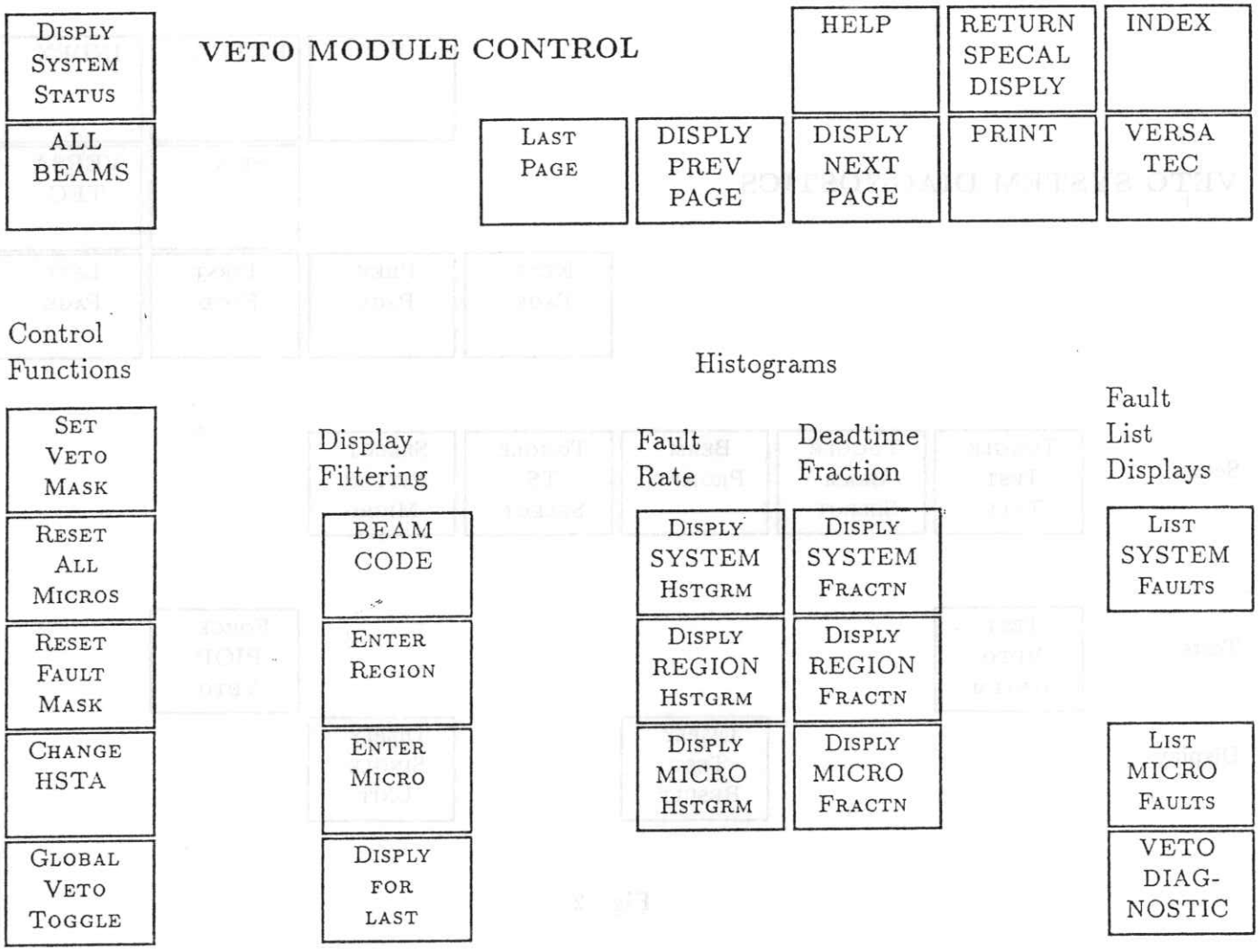

DISPLY SYSTEM STATUS
VETO MODULE CONTROL
HELP
RETURN SPECAL DISPLY
INDEX
ALL BEAMS
LAST PAGE
DISPLY PREV PAGE
DISPLY NEXT PAGE
PRINT
VERSA TEC
Control Functions
Histograms
Fault List Displays
SET VETO MASK
Display Filtering
Fault Rate
Deadtime Fraction
RESET ALL MICROS
BEAM CODE
DISPLY SYSTEM HSTGRM
DISPLY SYSTEM FRACTN
LIST SYSTEM FAULTS
RESET FAULT MASK
ENTER REGION
DISPLY REGION HSTGRM
DISPLY REGION FRACTN
CHANGE HSTA
ENTER MICRO
DISPLY MICRO HSTGRM
DISPLY MICRO FRACTN
LIST MICRO FAULTS
GLOBAL VETO TOGGLE
DISPLY FOR LAST
VETO DIAG-NOSTIC

Fig. 1

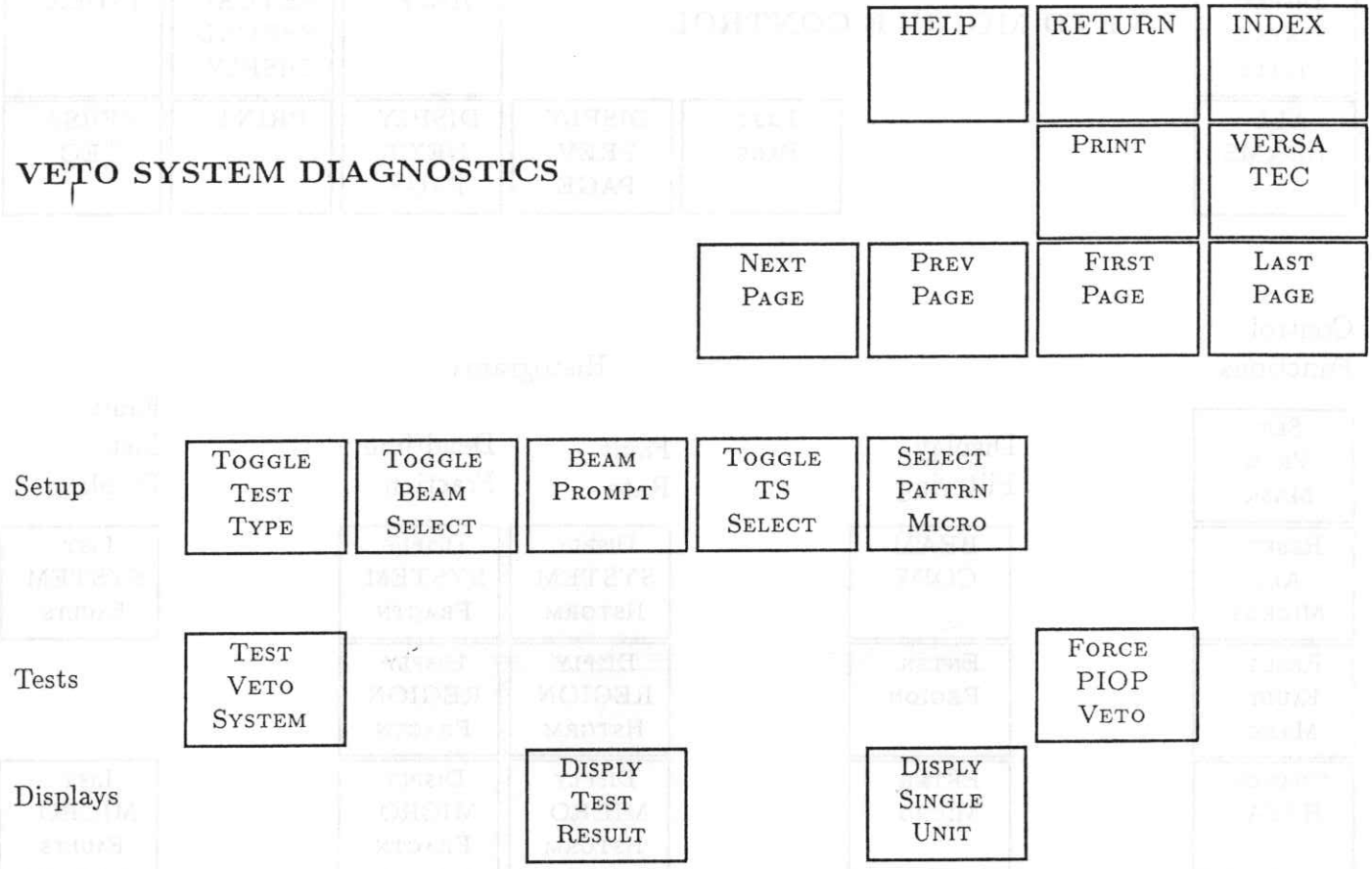
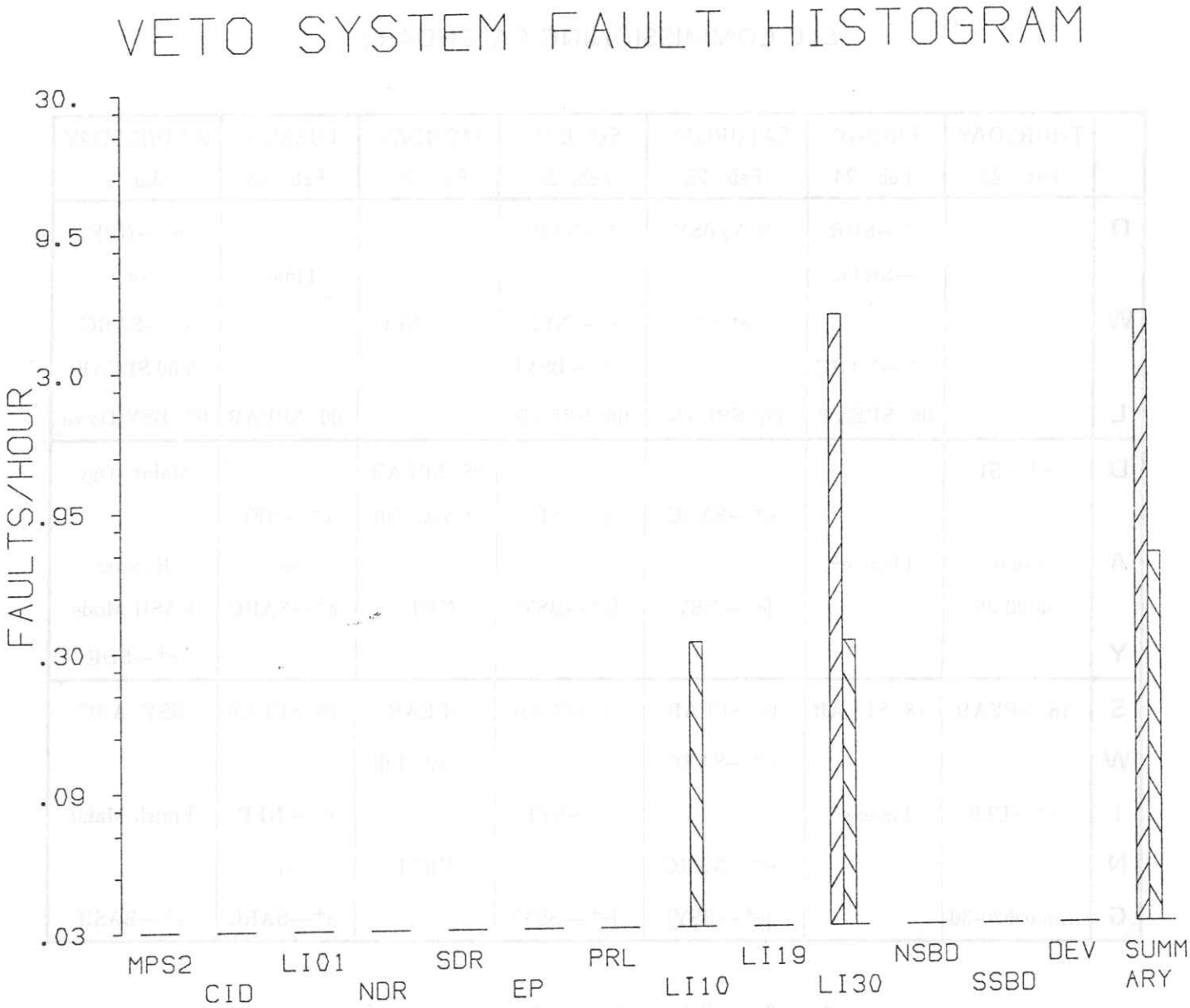

HELP
RETURN
INDEX
VETO SYSTEM DIAGNOSTICS
PRINT
VERSA TEC
NEXT PAGE
PREV PAGE
FIRST PAGE
LAST PAGE
Setup
TOGGLE TEST TYPE
TOGGLE BEAM SELECT
BEAM PROMPT
TOGGLE TS SELECT
SELECT PATTRN MICRO
Tests
TEST VETO SYSTEM
FORCE PIOP VETO
Displays
DISPLY TEST RESULT
DISPLY SINGLE UNIT

Fig. 2



FAULT RATES

	MPS2	CID	LI01	NDR	SDR	EP	PRL	LI10	LI19	LI30	NSBD	SSBD	DEV	SUMMARY
SHORT	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	4.58	0.00	0.00	0.00	4.58
LONG	0.00	0.00	0.00	0.00	0.00	0.00	0.00	.312	0.00	.312	0.00	0.00	0.00	.624
LONGER	0.00	0.00	0.00	0.00	0.00	0:00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

22-FEB-89 23:30:14

Fig. 3

SLC COMMISSIONING CALENDAR †

	THURSDAY Feb. 23	FRIDAY Feb. 24	SATURDAY Feb. 25	SUNDAY Feb. 26	MONDAY Feb. 27	TUESDAY Feb. 28	WEDNESDAY Mar. 1
O W L		$e^+ \rightarrow$SDR $\rightarrow$SRTL $e^- \rightarrow$NARC 06: SPEAR	BSY/BSY e^+ e^- 06: SPEAR	$e^- \rightarrow$NARC ..... $e^- \rightarrow$NFF [$e^+ \rightarrow$BSY] 06: SPEAR	$e^- \rightarrow$NFF	Linac ϵ 06: SPEAR	$e^- \rightarrow$NFF or $e^+ \rightarrow$SARC 5:30 SPEAR 07: BSY Down
D A Y	$e^+ \rightarrow$S1 Linac Φ 20-30	Linac $e^\pm$	$e^+ \rightarrow$SARC [$e^- \rightarrow$BSY]	$e^- \rightarrow$NFF [$e^+ \rightarrow$BSY]	08: SPEAR Cont. Fill NRTL	$e^- \rightarrow$NFF or $e^+ \rightarrow$SARC	Maint. Day ..... Recover BASII Mode $e^+ \rightarrow$SDR
S W I N G	18: SPEAR $e^+ \rightarrow$LTR LinacΦ20-30	18: SPEAR Linac $e^\pm$	18: SPEAR $e^+ \rightarrow$SARC ..... $e^- \rightarrow$NARC [$e^+ \rightarrow$BSY]	18: SPEAR $e^- \rightarrow$NFF [$e^+ \rightarrow$BSY]	SPEAR Cont. Fill NRTL	18: SPEAR $e^- \rightarrow$NFF or $e^+ \rightarrow$SARC	BSY, ARC Finish Maint. $e^+ \rightarrow$BASII

$e^+ < 1.2 \times 10^{10}$ beyond Sector 30.

$e^- < 1.2 \times 10^{10}$ beyond Sector 30.

Linac pulses at 60 Hz; 30 Hz to arcs.

† This calendar is provided for informational purposes only. Neither the Software Engineering Group nor the SLC management accept any responsibility for its accuracy. Schedule subject to change without notice.