Swapper makes several tests to ensure that programs are not unnecessarily placed on the big drum and that, when a program is swapped onto the big drum, it is the worst in the system. The simplified explanation of these test is given below in the order that the tests are made; these steps are also keyed to the accompanying block diagram (Fig. 4). Note that in each case other than running a user program the Swapper is then restarted to perform another evaluation.

1. If there is an unused core, the best accessible program (BAP) is swapped in and nothing is swapped out.

2. If the BAP is on the little drum, then the worst program in core is swapped with the BAP.

3. The BAP is now known to be in limbo or on the big drum; the Swapper will look for an unused field on the little drum. If there is an unused field, the worst user in core will be swapped onto that field before bringing in the BAP. This procedure may require the Swapper to wait for the little drum to complete some other operation, but, even if this is so, the program will be swapped out more rapidly than if it were swapped onto the big drum (and it can be retrieved much more rapidly).

4. At this point, the little drum is known to have no unused fields, so the only way to create space in core for the BAP is to swap some program onto the big drum. If the BAP is in limbo and the Swapper has already assigned some program to be swapped on the big drum, then all programs in limbo will be considered inaccessible and the Swapper will be restarted to find a new BAP. To understand the reason for this procedure, recall how a big-drum swap is initiated. The program to be swapped out of core is started in Executive core and executes instructions to link it to the I-O Processor. In order for this program to be started, it must be the BAP, and it won't be the BAP if the programs in limbo are ac-