

TRACK is a cost effective testing and debugging system for CICS programs that enables fast and effective program testing during both development and maintenance of CICS applications, improving programmer productivity and encouraging the production of more resilient programs.

Track's features include:

- A menu based command structure
- Point and click techniques for navigation and display
- Program Monitoring that provides: transaction abend interception, storage protection, wild branch detection, and program loop detection
- Source level debugging with automatic display of the variables referenced by the current statement
- Conditional and unconditional user defined halt points
- Display and modify data in storage
- Define items to be displayed on demand or at each program stop point
- Debug statically linked or dynamically called subprograms
- Alter program execution path (skip around or intentionally test specific code, step through program statements)
- Debug programs running on one terminal device, while showing Track displays at another
- Debug programs running in background tasks unattached to any terminal
- Control scope of monitoring ranging from all programs through to individual selected programs
- File and temporary storage record creation, display and alteration
- Security sub-system to permit precise control of commands that programmers are permitted to use



- Audit trail to log details of changes made during debugging

Track Benefits

- Improved programmer productivity from CICS test sessions by permitting the resolution of multiple problems in a single session
- Reduced development times due to improved resolution of problems
- Better tested and more resilient programs
- Ability to test infrequently used logic paths to identify and correct obscure program errors
- Reduced dump analysis
- Improved CICS system stability with better protection against storage corruption than available with native CICS
- Improved programmer morale
- Reduced overall cost from significantly improved productivity and fewer production outages due to reduced error levels

Track Features in Detail

Menu Based Command Structure

All **TRACK** facilities are accessible via clearly informative menu screens. Each menu provides an indication of the functions available to allow easy use of them. Options may be selected from a menu by positioning the cursor on the appropriate line and pressing the ZOOM PF key or by typing the function number shown on the left of the option description into the command field that is present on every TRACK screen. Each line on the menu also shows the fast path id associated with the option. These fast path ids can be used by more experienced users to navigate directly to any display from any point in the system.

Point and Click

Wherever feasible commands are effected by positioning the cursor and then pressing an appropriate function key to cause a command to be executed. This technique is used on all menus to select a function, on source displays to select variables for display, on source and link data directories to select source or link data for display and on monitor control displays to select items for expansion or deletion.

Program Monitoring

Provides abend interception, storage protection, wild branch detection and loop detection. Any program check or abend request in a monitored program results in a stop display at the point of the abend. Similarly any attempt to change storage that is not legitimately allocated to the transaction results in a stop as does an attempt by the program to branch to a random point. Limits may be set on the number of instructions that may be executed or the number of CICS calls that may be issued, between program halts. If a limit is exceeded, an automatic halt occurs. This allows the detection of program loops whether or not the loop includes CICS calls. Setting the CICS call limit to one can be used to produce a halt at every CICS call.

Source Level Debugging

The programmer's source code is captured at compile time and stored on the TRACK source file. It can then be displayed on-line and used for the setting of halt points prior to executing of the program as part of a CICS transaction. Halt points are set and reset by simply typing a one character command on the statement number of the displayed source.

When a stop is encountered, the source at that point is displayed together with the value of variables referenced in the current statement. These values can be altered by simply typing the new value. For COBOL programs an expanded view of data structures can be displayed using the GROUP command. This gives a fully formatted display including all data types and subscripted areas.

Halt Points

Program execution may be suspended at programmer-defined halt points. These may be unconditional or dependant on some logical condition. When a halt point is reached a stop for a 'halt request' is presented giving details of the current statement and variables referred to by the statement.

Unconditional halts always cause the program to be stopped and are easily set by typing a one character command on the appropriate program statement when the source is displayed. They can also be set using an Add Halt Point display where certain specialized halt offsets such as START or ENTRY can be used.

Conditional halts are also easily set by typing a one character command on the appropriate program statement. For a conditional halt however, a secondary screen is displayed to enable the condition to be provided. This is normally in the form of a comparison but may also include limits to the number of times a halt will occur and how often the halt will occur.

When any halt occurs, all Track debugging facilities are available for the examination and alteration of data (in storage, temporary storage, or on file) and of program code. This enables errors to be detected and corrected or changes made to ensure execution of particular sections of code. After each halt the programmer may terminate the transaction, produce a dump, continue execution normally, STEP through the program by statement or instruction, or even continue from a different point in the program.

Define Items to Display on Demand

Storage areas or program variables may be selected to be displayed at each program stop by defining them as Keep items. Once defined a keep item may be displayed in a data window on the Stop Display or in response to the K (keep) command. The items may be displayed either by value (i.e. in accordance with the data definition for the item) or in hex. There is no

limit to the number of items that may be defined in this way.

Display and Modify Data in Storage

Storage areas or program variables may be displayed and altered singly or collectively in a number of ways. The display may occur as the result of a direct display command, as the result of a variable being acted upon by the current program statement, as the result of a Group variable display command or a Keep item display command. However, the data display originates, the data displayed may be changed simply by over-typing the existing value. If data is displayed in hex and character format, changes can be made to either form of the display. This makes it possible to assign incorrect values to areas to help test data analysis logic. When displayed data cannot be changed for any reason, the displayed data will be protected to prevent changes.

Debug Subprograms

In the situation where multiple components of a program are to be debugged simultaneously, TRACK permits halt points to be established in any component of the program whether statically linked to the main program or called dynamically. For statically linked components it is important that the link map for the program is stored on the source file to enable the names and locations of each component to be established.

Alter Program Execution Path

At a program stop for whatever reason execution can be continued at a point other than that at which the program was last halted. This allows the bypassing of erroneous code, the simulation of conditions which are hard to create and the testing of infrequently used logic paths.

Single-Stepping

At a halt, program execution may be continued by instruction stepping with a further halt after each step. A step may be one or more program statements or machine instructions. It is also possible to step immediately to a specified location in a halted program. Stepping is normally confined to the current program module to avoid stepping through compiler generated logic needed to handle complex statements. However, the scope of the stepping operation can be widened to encompass the whole program or multiple programs.

Program Flow Analysis

TRACK maintains a table of the most recent branches taken during program execution. At any stop point the flow command can be used to display this information to show the program's recent execution path and from that it is possible to determine why the program reached the stop point and whether it has reached that point correctly or as the result of a logic error. The display identifies the 'from' and 'to' statement numbers when source data is available and the names of called modules when link edit data is available.

Debug Programs at a Separate Terminal

Since CICS supports a range of devices, some of which do not have display capabilities, TRACK permits the establishment of debug relationships so that a program running on one device can be debugged from another terminal. The same applies to programs that run as part of non-terminal tasks such as those that process requests received from remote devices and systems.

Files and Temporary Storage

Records in CICS datasets and items in temporary storage queues may be viewed and changed and new records added. This enables test data to be created and changed and test results examined. This facility is also very useful for correcting data corrupted by a program that is under development to enable subsequent programs to execute successfully.

Security

TRACK has a powerful security feature that enables the system administrator to control who may use the TRACK system and which facilities they may use. It is possible to deny individuals the use of the file access facility or restrict access to certain files and permit read-only access. Similarly, the use of various debugging commands can be denied or restricted to display only. Access to certain data and changes thereto can be made subject to password protection with passwords unique to each user.

Modification Log

Any changes made using TRACK to storage areas, files, or temporary storage queues are logged to a CICS transient data destination. The log identifies the individual that made the change and identifies the time at which the change was made and the terminal used to make the change.

Bits Software Ltd.

Track Benefits in Detail

Faster Debugging

Track provides faster debugging of program logic, coding, and data errors. It provides for user halts at the transaction level or in any sub-routine and halt of program execution at programmer defined halt points that are activated only when certain conditions are encountered. Data can be displayed using variable names. Tasks executing at another terminal or printer or unattached to any terminal can be debugged. Loops can be trapped by setting instruction and CICS call limits.

Faster Corrections

Track allows single stepping through a program to follow the program logic. This can be done by either single machine instruction or by program statement. It also pinpoints errors for you on the screen. COBOL, PL/1 and Assembler source can then be displayed as well as data files. Corrections are made interactively and execution of the program then continues. Program flow after a halt can be redirected to test infrequently used logic paths.

Faster Turn Around

Multiple errors can be examined and resolved in one execution of the program. No more waiting to resubmit the program for additional compiles, doing more testing, and looking at more dumps.

Reduced Dump Analysis

It is no longer necessary to produce core dumps for each program error. Track points out exactly what you need to know on the screen. It takes the drudgery out of debugging and testing.

Stable CICS Environment

Track can also be used to monitor any specific program running under CICS. It detects program abends or illegal CICS operations. Track protects CICS from transactions causing table storage violations, thus preventing CICS crashes. Track uncovers those intermittent, hard to find bugs, in both a testing and production environment.

Supported Environments

VSE/ESA, z/VSE: all releases
OS/390, z/OS: all releases
CICS: 1.7 - TS 1.1 (VSE) or TS 3.1

Languages: Cobol, Cobol II, COBOL for MVS/VSE, Enterprise COBOL, PL/1 for MVS/VSE, Enterprise PL/1, and Assembler

Pricing

Track is competitively priced. Most equivalent products are 3 to 5 times the price. TRACK upgrade charges are minimal whereas other products often result in large upgrade fees when upgrading your operating system release or swapping out a CPU. Sometimes, these fees apply even if the capacities of the old and new CPUs are the same!

Free Trial

See the benefits yourself with a no obligation free trial as Track is easy to install and use with no modifications to CICS required apart from the products program, transaction and file definitions.

Vertrieb für Deutschland - Österreich und Schweiz (deutschsprachig):

Herbert Sund

Unternehmensberatung - Softwarevertrieb
Vor der Grube 4

64331 Weiterstadt

Tel.: +49 - 61 50 - 1 77 51

Fax: +49 - 61 50 - 4 05 34

Email: HSund@T-Online.de

