MAGNUS
ENGINEERING & MAINTENANCE LTD.

# MCTC

# PRINCIPLES

# OF

# OPERATION

35 Hayarkon St., 51204 Bnei-Brak, ISRAEL Tel: 972-3-5780480 Fax: 972-3-5780525
e-mail: info@magnus-eng.com

# Table of Contents




35 Hayarkon St., 51204 Bnei-Brak, ISRAEL Tel: 972-3-5780480 Fax: 972-3-5780525
e-mail: info@magnus-eng.com

## 1. GENERAL

Magnus Computerized Test Cell system (MCTC) is a powerful **generic** software package that provides all the data acquisition and data processing functionality required for the operation of automated engine test cells.

The rest of this section and the following paragraphs provide a broad description of the overall capabilities of the MCTC software package. Specific implementations utilize a subset of these capabilities to generate a system which meets the monitoring, control and data processing needs of specific test cells and specific engine types.

For some engine types, the generic software package is complemented with engine specific software code that performs engine-specific complex data processing functions. The user and run summary reports may also be customized to meet specific users' needs.

Figure 1-A: MCTC implementation for F100-PW-229 engine.

## 1.1. System Benefits

The implementation of a computerized Test Cell system in general and MCTC in particular provides significant benefits to any organization which maintains and tests engines.

- **Improved engine reliability**: By ensuring complete and accurate execution of test procedures, MCTC increases the reliability of engines that have been successfully tested in the test cell. Automated validation of required engine behavior/performance provides additional level of reliability by eliminating the possibility of human errors.

- **Increased testing efficiency**: MCTC can perform any required computation almost instantaneously. Using the extensive set of functions provided by the Test-Builder facility, test procedures can be optimized so that they will be executed in a minimal period of time – thus allowing the most efficient (yet complete) execution of the required tests. The increased efficiency means **faster turnaround time** as well as **fuel savings** (by minimizing time of maximum-power engine operation).

- **Prevention of operational mistakes**: By including proper instructions to the operators, and a matching set of automatic tests, the testing procedure can reduce the risk of an operational mistake being made by a non-experienced operator. If such a mistake does occur, appropriate additional instructions and checks may be added by the user (with no need of any additional programming!) to prevent similar future events.

- **Avoid engine limit violations**: MCTC provides continuous engine limit-violation tracking. This set of checks can be easily expanded to include early warnings – which will alarm the operator before an engine limit has been violated (e.g. if an engine may operate above a given temperature for 12 seconds, set ALARM After 4 seconds above limiting temperature). This will **prevent engine damage**, and will **eliminate unnecessary maintenance activity**.

- **Avoid unnecessary engine runs:** By recording the entire run session, and combining this data with MCTC powerful playback and analysis tools, MCTC allows the user to extract maximum information from any test session. In many cases this will enable the user to fully analyze the source of any problem detected during the run – thus eliminating the need for additional test runs.

- **Single focus of attention:** By allowing complete integration of engine and test cell facilities data (monitoring and control functions) into a single system, MCTC enables the operator to focus his/her attention on a single system (in comparison, in other test cell set ups the operator must concurrently pay attention to many different systems/displays). This feature enhances **easier** as well as **safer test cell operation**.

## 1.2. Hardware Implementation Schemes

MCTC modular design enables two basic implementation schemes:

- A simple and compact implementation where all the functions of the system are carried out on a single computer.

- A complex, expanded implementation where the system's functions are distributed between two independent but coordinated computers:

- **DAC** (Data Acquisition Computer): performs basic data acquisition functions.
- **MTC** (main Test Cell computer): performs all other test cell functions.

The first implementation scheme is used for relatively simple engines with a small number of tracked parameters (e.g. diesel engines) whereas the second implementation scheme is used for more complex engines with many tracked parameters and complex data processing requirements (e.g. turbo-jet and turbo-fan engines).

The system capabilities can be further extended in a modular fashion by adding one or more of the following optional components:

- **ESMC** (Engine Shop Manual Computer): This computer displays the original vendor testing instructions. The MTC will automatically instruct this computer to display the applicable section of the vendors' documentation according to the current test step.

Figure 1-A above depicts the control room of a test cell with a complex MCTC implementation including DAC, MTC and ESMC systems.

- **RVC** (Remote Viewing Computer): An installation of MCTC on a remote PC connected to the MTC via standard network interface (Internet or intranet) which allows the remote user to view the engine run in real time and view, collect and analyze the run data independently of the test cell computers. (An on-line chat with the test cell operator is also provided).

- **OADC** (Offline Analysis and Development Computer): An installation of MCTC on a remote computer that may be connected to one or more test cell computers (MTC-s). This component allows a test cell engineer to perform in-depth analysis of test runs from a number of different cells, and to develop and change testing procedures for these test cells (includes a facility to distribute newly developed definitions to connected test cells in a controlled manner).

## 1.3. Multiple Levels of Testing Automation

MCTC supports a wide range of testing automation levels:

- **Monitoring**: MCTC Acquires, displays and stores all engine and test cell parameters. User performs "unguided" test procedures.

- **Test-Procedure Tracking**: MCTC "guides" the user through the predefined test procedures and performs automatic checking for successful engine operation. Actual engine control operations are executed manually.

- **Automatic Testing**: MCTC controls the engine and the external facilities (e.g. dynamometer, fuel heating system etc.) and executes automatically predefined test procedures.

## 1.4. Supported Platforms

MCTC runs on IBM-PCs and compatibles. MCTC has been implemented up to date on the following operating systems:

- Microsoft's MS-DOS™ 6.22
- Microsoft's Windows For Workgroup™ (WFW) 3.11
- Microsoft's Windows NT™ Workstation 4.0
- Microsoft's Windows XP™ Professional

## 1.5. Development Tools

MCTC (Windows versions) was developed using the following tools:

- Microsoft Visual Basic™.
- Microsoft ACCESS™.

Additional third party components include (partial list):

- BPS Graphic Server Version 5.0
- XCEED Zip Compression Library©.
- National Instruments NI-DAQ™ Driver package
- National Instruments NI-488.2™ Dos/Windows for AT-GPIB/TNT
- DIGI International DIGIBOARD™ Intelligent board driver

- IOTECH VISUALAB GUI©.

### 1.6. Software Quality Assurance

MCTC was developed under the guidelines of MIL-STD-2167. The development process as well as the on-going maintenance of this software package is accompanied by a software quality assurance program under the guidelines of DOD-STD-2168. The quality assurance procedures and their implementation has been reviewed and approved by various external agencies (Israel Standards Institute – ISS, Israel Air Force – IAF, P&W software quality assurance group).

### 1.7. Connectivity to External Data Acquisition Systems

MCTC has been designed to allow seamless integration of data acquired from many different external sources into a unified data analysis, storage and display process. MCTC supports a large number of standard buses over which data from external subsystems may be received. The incorporation of a new external data acquisition subsystem into the MCTC framework is a very simple task.

The following external data sources are currently supported by MCTC (partial list):

| Bus | Sample Applications | Sample Applications |
|---|---|---|
| On-Board plug-in | National Instruments (any NI-DAQ supported card) | PC-LABCARD PCL-730 32 channel DIO card |
| GPIB (IEEE-488) | TRIGTEK Model 645A Vibration Monitor/Analyzer | |
| RS232 | GE (F110) EMSC UART | SCAIME DMJ (thrust measurement) |
| RS422 | P&W (F100) EDU/DEEC UART | |
| RS485 | GANTNER E.BLOXX | Rosemount MICROMOTION (mass flow meter) (MODBUS) |
| 1553B | P&W (F100) DEEC | GE (F110) EMSC |
| ARINC 429 | APU controller-PWC<br>PW4000 EEC<br>JT9D EEC | CFM56-7 EEC |
| CANBUS | MTU TCS and CDS/CR | |
| Ethernet | GANTNER EGATE (data hub)<br>PLC-Direct<br>PSI (pressure scanners) | TRIGTEK 686B (Vibration Analyzer).<br>PBS4100 (Vibration Analyzer) |

## 1.8. Current Installations

MCTC has been implemented for the following engine types (partial list):

| Diesel/Piston | Turbo-Jet | Turbo-Shaft/Prop |
|---|---|---|
| MERCEDES-OM355 | F404-GE-100D | T700-GE-700 |
| DEUTZ-F12L-413 | J85-GE-21B/C | T700-GE-701 |
| CUMMINS | F110-GE-100 | T700-GE-701C |
| CUCV | F100-PW-100 | CT-7 |
| MAN-16240 | F100-PW-220 | T64-T4C2 |
| DEUTZ-F8L-413 | F100-PW-200DPI | MAKILA 1A1 |
| MAN-40-440 | F100-PW-229 | T-56-A15 |
| | J52-PW-P8 | Tyne-MK22 |
| ROTAX-914 | J52-PW-P408 | PT-6A Series |
| AR801/802 | PWC-535 | PT-6T Series |
| | CFM56-7B | |
| | CFM56-3 | |
| | CFM56-2 | |
| | JT3D | |
| | PW4000 | |
| | JT9D | |
| | JT8D-200 | |

## 2. DESIGN PRINCIPLES

The following principles guided the design of the MCTC system.

### 2.1. Ease of Operation

The MCTC system is designed for simple and easy use. Detailed pop-up menus and "soft" switches direct the user through selection of available options.

Standard color coding scheme (messages, menus, and parameters are color coded) simplifies system operation.

MCTC design allows for easy adaptation of the user interface to any desired language. Currently English, Turkish and Hebrew versions exist. The system supports bilingual applications (e.g. system language can be changed from English to Turkish on-line).

### 2.2. Robustness

The system protects itself against user errors as well as hardware malfunctions (e.g. noise on input communication lines, failure of external data acquisition devices etc.) and will automatically recover from such errors and continue to function properly.

In the 2-computer implementation, the Data Acquisition computer (DAC) can run independently of the Main Test Cell Computer (MTC) and will continue to operate if the MTC fails during an engine test run.

### 2.3. Dynamic Configuration

A major design goal for MCTC is to achieve the maximal separation between software code and the data definitions which are unique to a specific test cell and engine type. By allowing dynamic modifications of these data definitions, MCTC becomes a generic package which can be adapted very easily to different test cells and different engine types. The end user can modify the MCTC package to adapt it to ongoing changes in the test cell or in engines' vendors test instructions – with no need for additional software programming.

Additionally, MCTC is usually set up with (many) spare physical input channels in order to provide for future expansion in data acquisition capacity - again with no need for any additional programming.

MAGNUS ENGINEERING & MAINTENANCE LTD.

### 2.4. Continuous Monitoring For Limit Violations

MCTC has been designed to continuously monitor incoming data in order to recognize engine limit violations and engine or test cell faults. All recognized engine limit violations or faults are immediately displayed on the screen, and are also stored and archived as part of the historical run data file. The same mechanism is used to generate early alarms – which enable the operator to avoid actual occurrence of engine limit violation.

### 2.5. Standard Graphic User Interface

The MCTC runs under the familiar Microsoft Windows environment with the standard use of functional keys, icons and pointing device operation. As a rule, maximum usage is made of the pointing device (“mouse”) to facilitate easy and user friendly system operation. However, most “mouse” functions can be also executed with specific keyboard combinations to allow the experienced operator a quicker mode of operation.

## 3. System Functions

Figure 3-A describes the main menu of the MCTC functions.

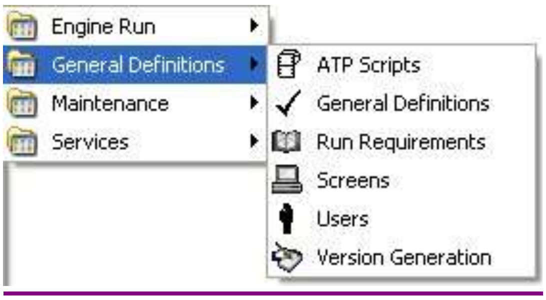
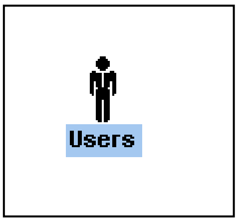



Figure 3-A: Main Menu of MCTC.

35 Hayarkon St., 51204 Bnei-Brak, ISRAEL Tel: 972-3-5780480 Fax: 972-3-5780525
e-mail: info@magnus-eng.com

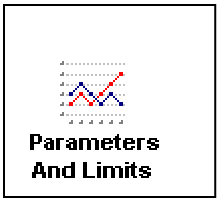
The system provides the following basic functions (key concepts and functions are described in more detail in the following sections):

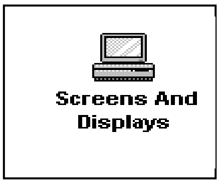
### 3.1. User Authorization Functions

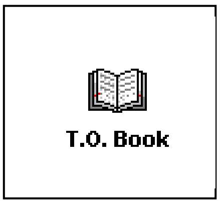


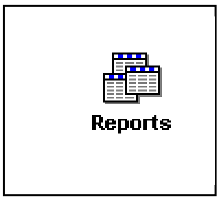

- Update list of authorized Users
- View/Report Contents of Tracked Events log file
- Erase Contents of Tracked Events log file.

### 3.2. Dynamic Test Cell Definition Functions



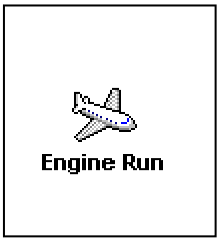




- Parameters
- Engine States
- Engine Limits
- Output operations
- Schedules/Graphic elements
- Parameters' Lists
- Faults/Warnings
- Control Functions (output operations)
- Safety Alarms
- Display Screen Layouts.
- Run reasons/Run Types
- Test Procedures.
- New version generation procedure
- Test Cell definition reports

### 3.3. Engine Run initiation Functions



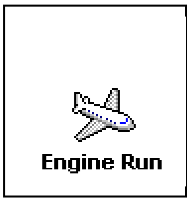

- Engine identification
- Run reason(s) identification
- Run type selection.
- Operator(s) identification.
- Modification of dynamic run time properties.
- Engine specific run initialization functions

35 Hayarkon St., 51204 Bnei-Brak, ISRAEL Tel: 972-3-5780480 Fax: 972-3-5780525
e-mail: info@magnus-eng.com

## 3.4. Engine Test Run Functions


- Data initialization
- Data acquisition
- Data Filtering
- Data storage
- Limit violation monitoring/treatment
- Engine Faults monitoring/treatment
- Real time Parameter Display
- Display of safety alarms
- User controlled recording modes (Normal/Transient/Static)
- On-Line modifications of Display Screens
- User-input data processing
- Test Procedure tracking
- Generation of System Parameters
- Automatic Test control (output operations)
- Manual test control (output operations)
- Event recording (Limits, Faults, Test-procedure events)
- Engine specific run time functions.

## 3.5. Engine Run Termination Functions

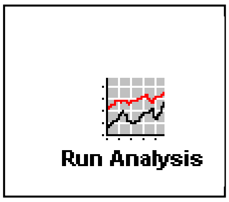



- Display of previous tests (run-continuations) results
- User defined run-continuation status
- User Defined final test status
- User defined remarks (free text)
- Generation of run reports
- Archiving of run data
- Engine specific run-termination functions

35 Hayarkon St., 51204 Bnei-Brak, ISRAEL Tel: 972-3-5780480 Fax: 972-3-5780525
e-mail: info@magnus-eng.com

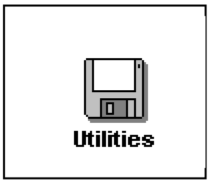
### 3.6. Post Run Analysis Functions

- Select and display historical runs based on various criteria.
- Full Playback of a saved run.
- Report Generation for a single saved run data.
- Graphs generation for a single saved run data.
- Report generation for a selected group of runs.
- Trending analysis (“health monitoring”).
- Exporting of run data into external formats (ASCI files, Excel spreadsheets) to enable connection to external analysis tools.



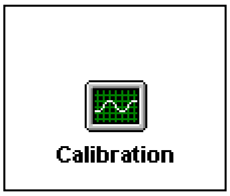

### 3.7. Off Line Archiving Functions

- Move Archived runs data files to Off-Line storage media.
- Restore Archived runs data from Off-Line storage media.



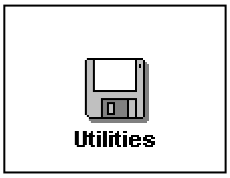

### 3.8. Test Cell Maintenance Functions

- Computerized Calibration Utility.
- Off-Line display of external systems’ data (e.g. PLC).




- Probes/Thermocouples configuration management.


### 3.9. MCTC Maintenance Functions




- Automatic daily backup of MCTC databases.
- Repair & Compact utility for MCTC databases.
- Optional restoration to any stored (older) version of the test cell definitions.
- Complete system backup (on removable magnetic media).
- System recovery procedure.

# 4. TEST CELL DEFINITION (DEF Utility)

## 4.1. Static definitions

For each implementation of MCTC, the following entities are predefined:

- External Data Sources for each engine type.

These entities cannot be modified by the end user (as any change here entails some hard-coded software changes). All the other entities which make up the test cell definition may be modified by the end user (including addition of a new engine type), as described below. A separate set of definitions is used for each engine type. Therefore the [authorized] user has to select a specific engine type in order to view and possibly modify the current definitions.

Figure 4.1-A displays the entry screen of the DEF (Definition) utility, with a list of engine types defined for the (sample) test cell.

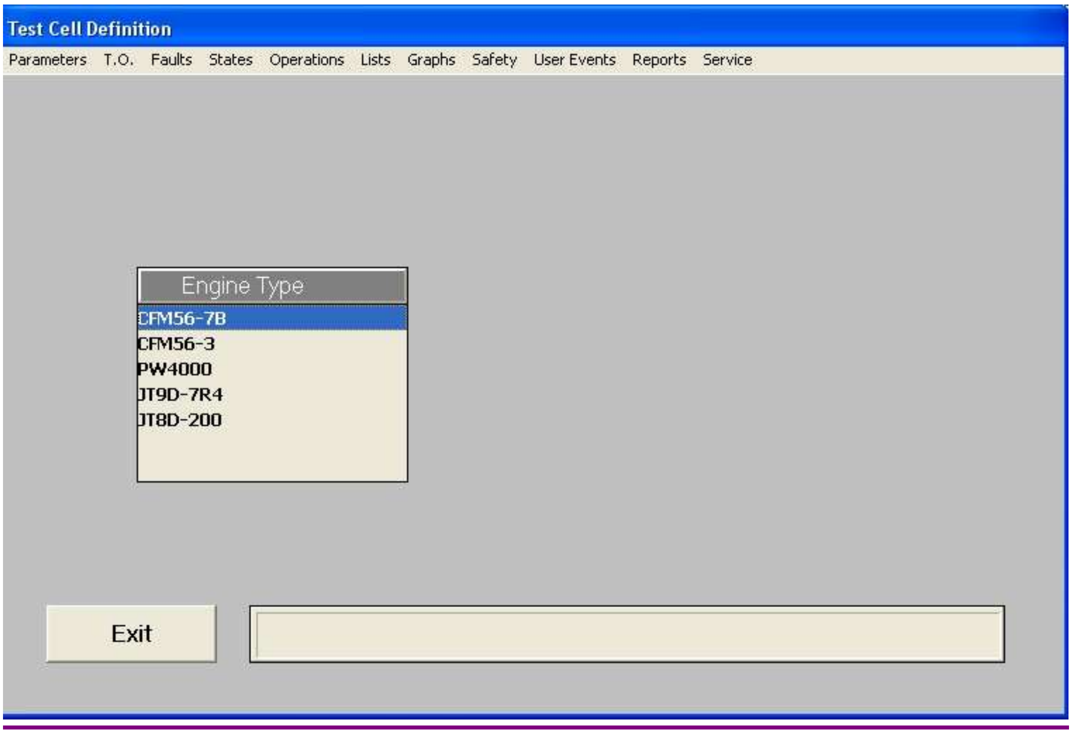



Figure 4.1-A: Test Cell Definition utility (DEF) entry screen.

## 4.2. Parameters

MCTC combines information from various sources into a single logical record. This record is called “a global scan” (or “scan”) and includes the values of all input parameters in a given point in time. The user can view and modify various attributes of the system parameters using the DEF utility. Figure 4.2-A displays the Parameters Update screen of the DEF utility.

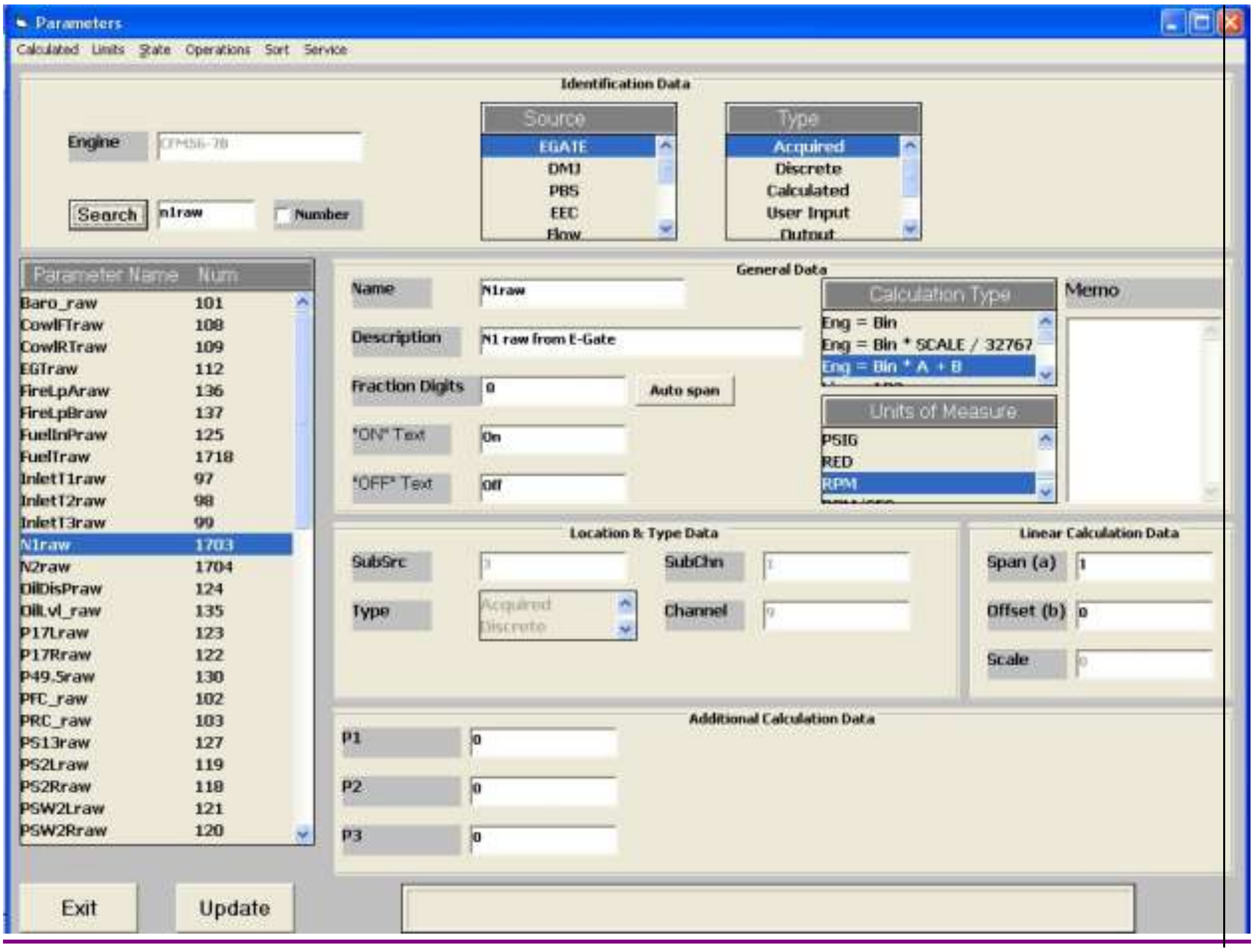



Figure 4.2-A: Parameter Update screen of the DEF utility.

MCTC handles parameters whose values are generated either externally or internally. The following parameter source types are supported:

### 4.2.1. Acquired Parameters

Acquired parameters are measurements of various engine, test cell or environmental phenomena which are acquired by external data acquisition subsystems and continuously transmitted into the MCTC.

Acquired parameters have the following attributes:

- External Source Identification
- Name
- Description
- Engineering units
- Physical address (slot, channel within slot etc.)
- Conversion function (from input format into engineering Units)
- Display accuracy.

As a general rule, all the channels of an external source (used as well as spare ones) are defined to the MCTC. This enables incorporation of new acquired parameters into the system without any additional programming.

#### 4.2.2. User Input Parameters

External values which cannot (for any reason) be acquired automatically by external subsystems can be entered into the system manually.

Example: For some engines, the LHV has to be defined to the system because it affects the engine performance calculations.

User input parameters values are updated in the "global scan" when the user manually updates their value. User input parameters are treated like regular (acquired) parameters except for their initialization:

> When starting a new Run-Continuation (2$^{nd}$ or more), User input parameters retain the value they had at the end of the previous run-continuation.

#### 4.2.3. System Parameters

Parameters whose value is determined internally by the MCTC during engine test run. System parameters values may be set by:

- Engine-specific hard-coded software routines.

- Test-Procedure tracking mechanism.
- Filtering, averaging and other generic MCTC mechanisms.

Examples:

- Maximum vibration value recorded during a specific test (“vibration sweep”) is stored by the Test-Procedure mechanism as a system parameter for later display/analysis.

- Transient time measured by the Test-Procedure tracking mechanism (e.g. – from IDLE to 85% power) is stored as a system parameter for later display/analysis.

- The raw (unfiltered) value of an acquired parameter which is being filtered by MCTC is stored by the Filtering algorithm as a system parameter for real time display and absolute limit violation checks.

As can be understood from the above examples, System parameters are not used exclusively for system purposes and are most often defined by the user in order to accomplish various tasks in the Test-Procedure tracking algorithms.

### 4.2.4. Output (Control) Parameters

Output parameters define external physical addresses that may receive (from the MCTC) various values which control the operation of the test cell or the engine. Values of the output parameters may be set manually by the user or automatically by the MCTC as the result of output operations which may be initiated by one of the following:

- Test-procedure tracking instruction.
- Detected engine limit violation.
- Detected engine fault.

Examples:

- Moving the throttle to a predefined target setting during automated testing.
- Activation of an audible alarm (BUZZER) by setting ON a bit in the PLC.

### 4.2.5. Derived (Calculated) Parameters

MCTC provides a powerful computation tool which allows the user to define new functions based on the acquired data. The complexity of the available computations is virtually unlimited.

Derived parameters value is the result of computations carried out on other parameters and other system entities. Derived parameters have the following attributes:

- Name
- Description
- Engineering units
- Conversion function (from internal storage format into Engineering Units)
- Display accuracy.
- Computation formula.

The computation formula for a derived parameter is defined in a "c" like format.

Operators can be:

- Arithmetical functions.
- Logical functions
- Graphical functions (e.g. interpolate "z" value from a 3-dimensional graph given "x" and "y").
- MCTC-specific functions (e.g. IF-THEN-ELSE, SPREAD, Linear Best-Fit etc.).

Operands can be:

- Any other parameter (Acquired, User input, System, Derived)
- Arithmetic (numerical) constants.
- Logical constants (TRUE, FALSE).
- Graphs/schedules.
- Engine states.
- Values returned by engine-specific hard-coded routines.

<u>Example</u>:

Figure 4.2-B displays the formula used to calculate the parameter "MOP" which is defined to be the difference between the Oil pressure reading (OilP) and the sump pressure reading (SumpP).

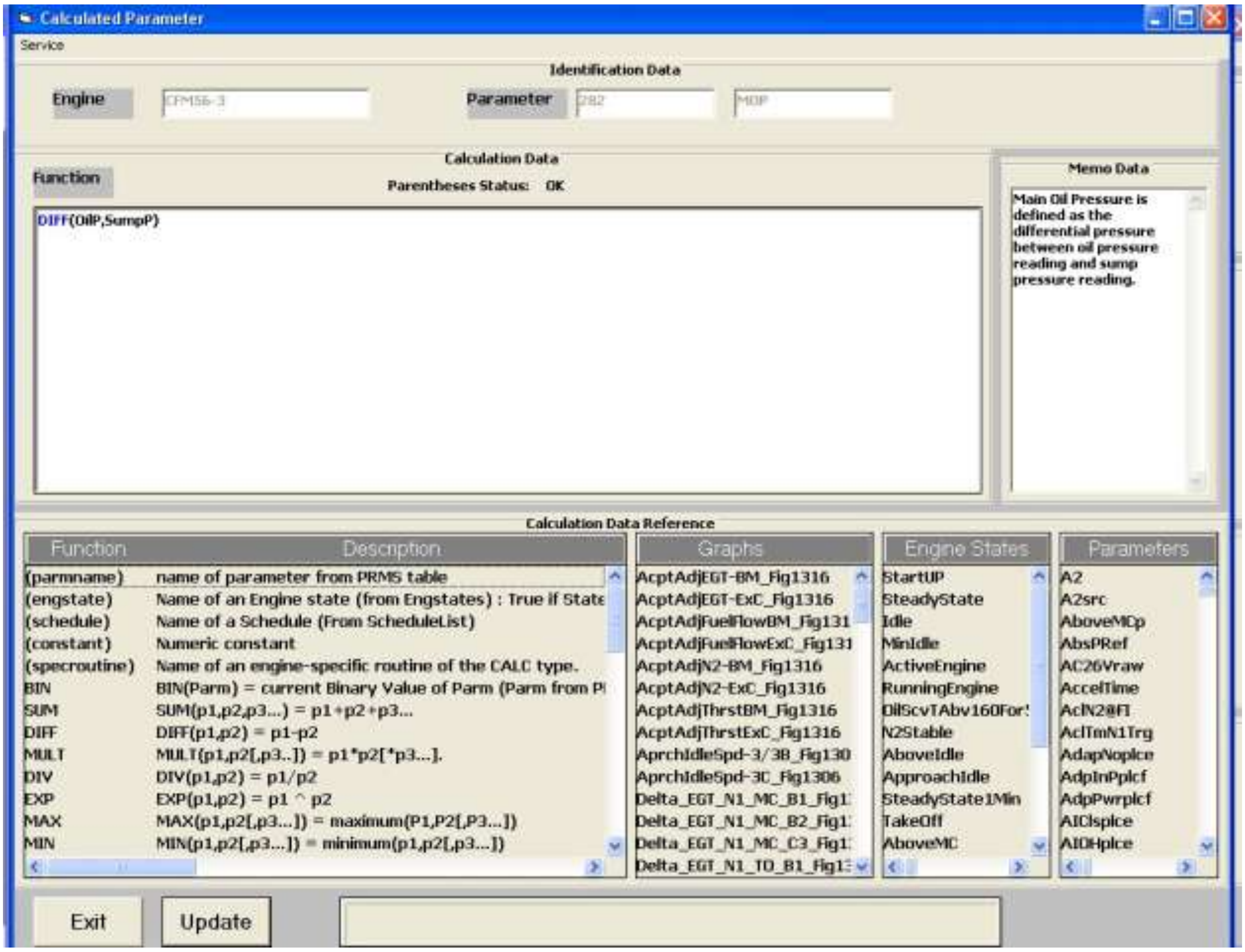



Figure 4.2-B: Calculated parameter formula definition screen.

Derived parameters are not stored with the other parameters in the "global scan". Instead, they are calculated whenever they are needed based on the current values of the other parameters (and the current engine states). This feature enables the user to define **new** derived parameters after a test run was completed, and view or report the values of these new parameters in the Post-Run analysis functions of the system.

### 4.3. Engine states

The identification of various engine operational states (e.g. – Engine Start Up, Steady-State condition, engine in Primary or Secondary mode etc.) is critical for the limit violation tracking mechanism and for the test-procedure tracking mechanism. In MCTC, engine states can be identified either by an engine-specific hard-coded routine or by the general parameter tracking mechanism. In the latter case, the engine state is determined based on the values of a single parameter.

In the general case, the engine state is "TRUE" if for a specified period of time the specified parameter (which may be from any source, including derived parameters):
- Is within a specified range of values and
- Does not fluctuate by more than a given limit.

Example:

Figure 4.3-A defines a stable-torque condition for the T700 engine. The engine state "N2Stable" is TRUE if the core speed is in the range 0-14694 rpm and does not fluctuate by more than 30 rpm for 5 seconds or more.

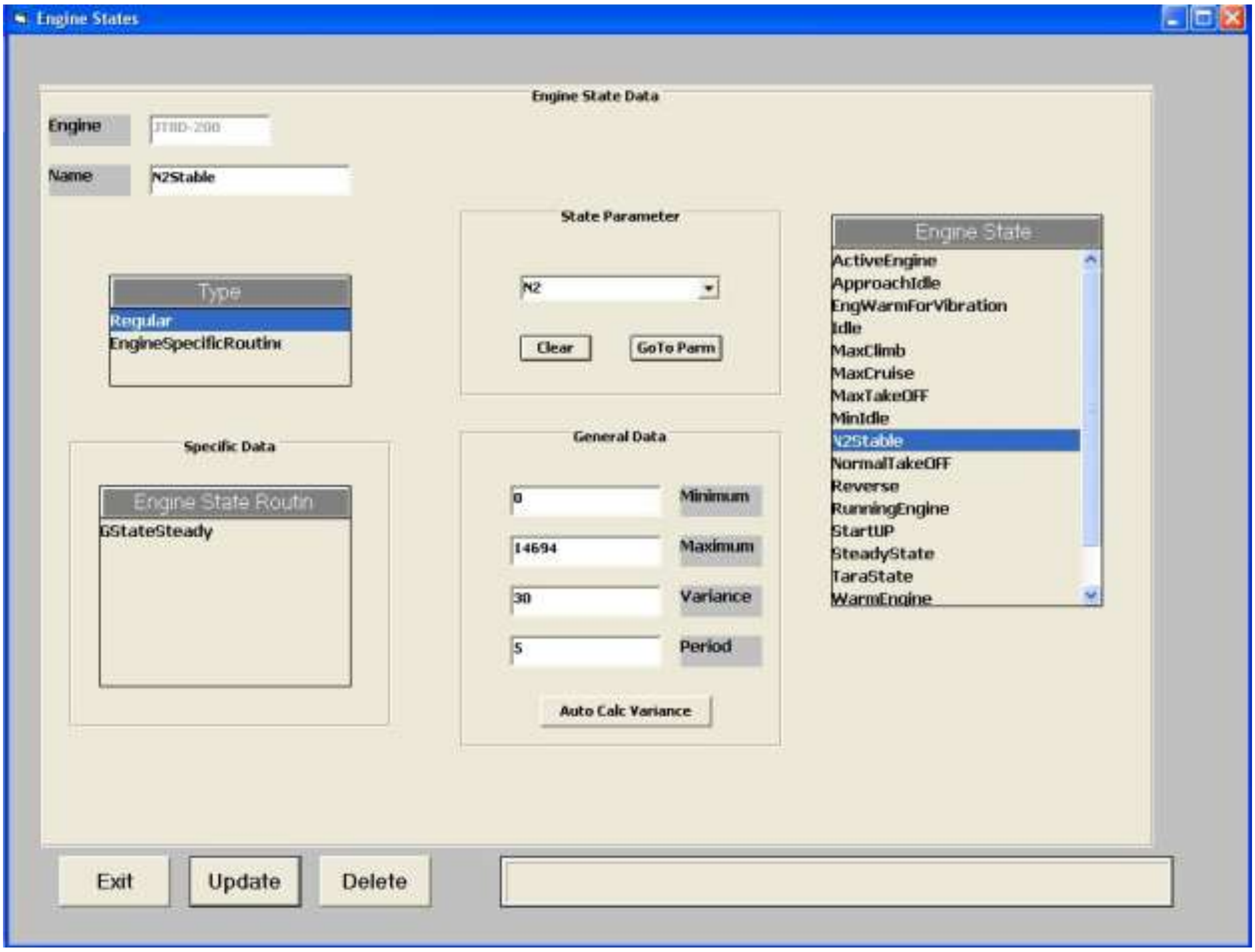



Figure 4.3-A: Engine state definition screen.

### 4.4. Engine Limits

MCTC handles engine limit tracking at 2 levels: simple limits and complex limits.

#### 4.4.1. Simple Limits

For each parameter (from any source) 6 simple limit-values may be defined:

- Maximum possible Physical value.
- Minimum possible Physical value.
- Maximum Limit value.
- Minimum Limit value.
- Maximum Warning value.
- Minimum Warning value.

These values are used during engine testing in the following ways:

- If current value of parameter exceeds the Maximum Limit value or is below the Minimum Limit value, The background color of the parameter is set to RED (Note: actual display is updated only if this parameter is currently displayed on either the DAC or MTC screen).
- If current value of parameter exceeds the Maximum Warning value or is below the Minimum Warning value, The background color of the parameter is set to YELLOW (Note: actual display is updated only if this parameter is currently displayed on either the DAC or MTC screen).
- If current value of parameter exceeds Maximum Limit value and action(s) are defined for this event – the specified action(s) will be executed.
- If current value of parameter falls below Minimum Limit value and action(s) are defined for this event – the specified action(s) will be executed.

Possible actions (one or more may be specified) include:

- Display of a RED warning message.
- Display of a GREY information-only message.
- Activation of an external audible alarm BUZZER.
- Abortion of the current Test-Procedure step.
- Activation of an output operation (other than BUZZER)

- Display of a Safety Alarm message.

35 Hayarkon St., 51204 Bnei-Brak, ISRAEL Tel: 972-3-5780480 Fax: 972-3-5780525
e-mail: info@magnus-eng.com

Note about background colors:

For some complex cases (e.g. – limit depends on engine state or value of other parameters) the mechanism described above is not sufficient. For these cases MCTC allows the definition of one (derived) parameter as the "color-defining-parameter" of another parameter. The background color of the second parameter is determined by the current background color of the first.

<u>Example</u>:

> Figure 4.4-A defines the simple limits for the "FuelInP" parameter of the JT8D-200 engine. No actions are defined for limit violations. Background color of "FuelInP" will be set to YELLOW if value exceeds 57 PSI and to RED if value exceeds 60 PSI.

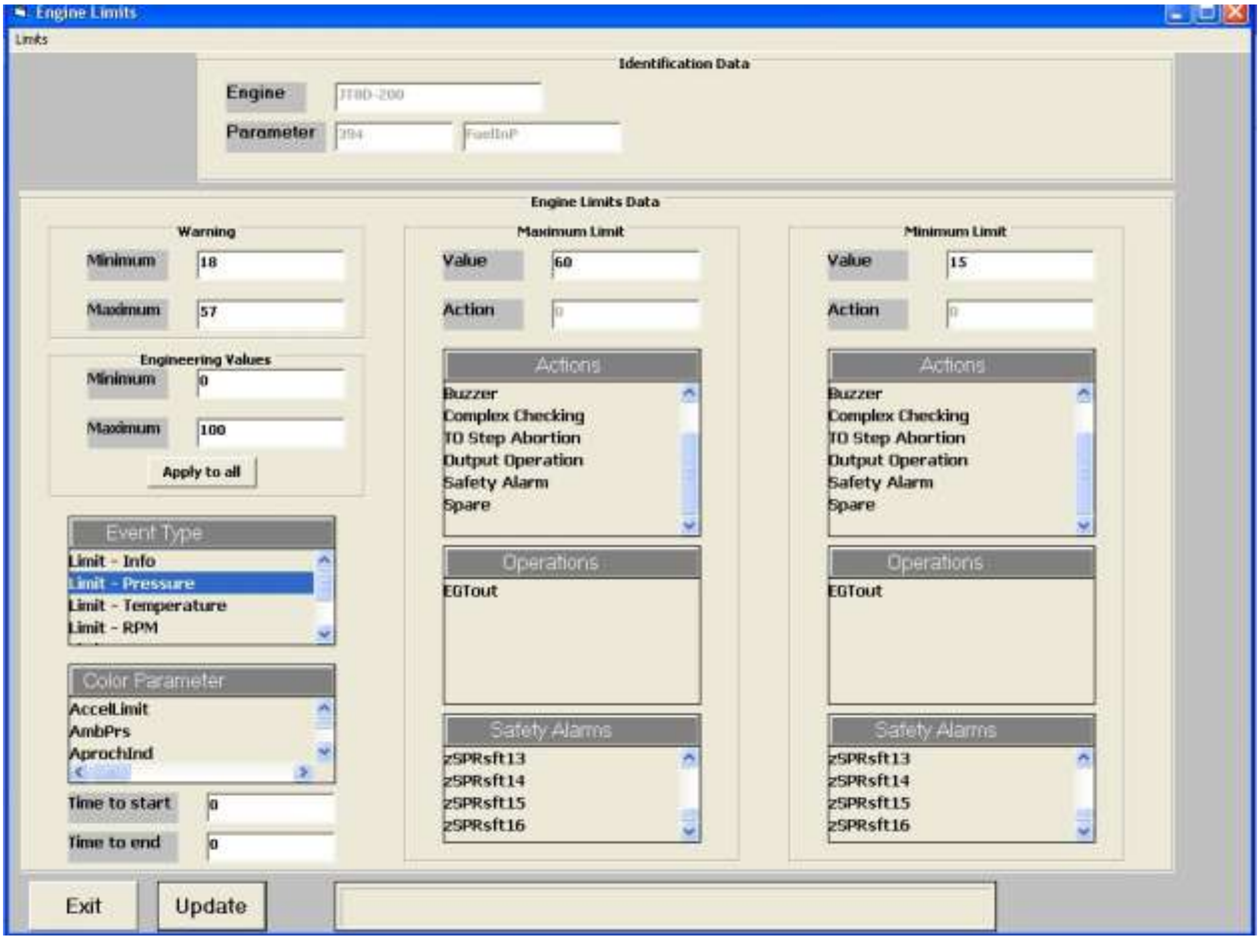



Figure 4.4-A: Simple Limit definition screen.

### 4.4.2. Complex Limits

Complex limits are used when simple limits are not sufficient to define the engine operational limits (e.g. – when the limit depends on engine states). Complex limits may be defined for any parameter (from any source including derived parameters). For each complex limit the following attributes are defined:

- Parameter to be checked.
- Name of Limit.
- Maximum Limit Value.
- Minimum Limit Value.
- Engine states (up to 3) which must be TRUE or FALSE to enable limit checking.
- Action(s) to be executed if Maximum limit is exceeded.
- Action(s) to be executed if value drops below Minimum limit.
- Duration during which Limit violation must exist before actions are executed (time filter).

The limiting value type may be:

- A numerical constant
- Another parameter (of any source)
- Value computed by an engine-specific routine
- Stability limit

Stability limit values are always numeric constants. The system monitors for stability limits only when the engine is stable ("STEADYSTATE" engine state is TRUE).

<u>Example</u>:

Figure 4.4-B defines the complex “FuelInP” limit of the JT8D-200 engine. If "FuelInP" exceeds 60 PSI while ActiveEngine state is ON, a RED alert message will be displayed to the operator.

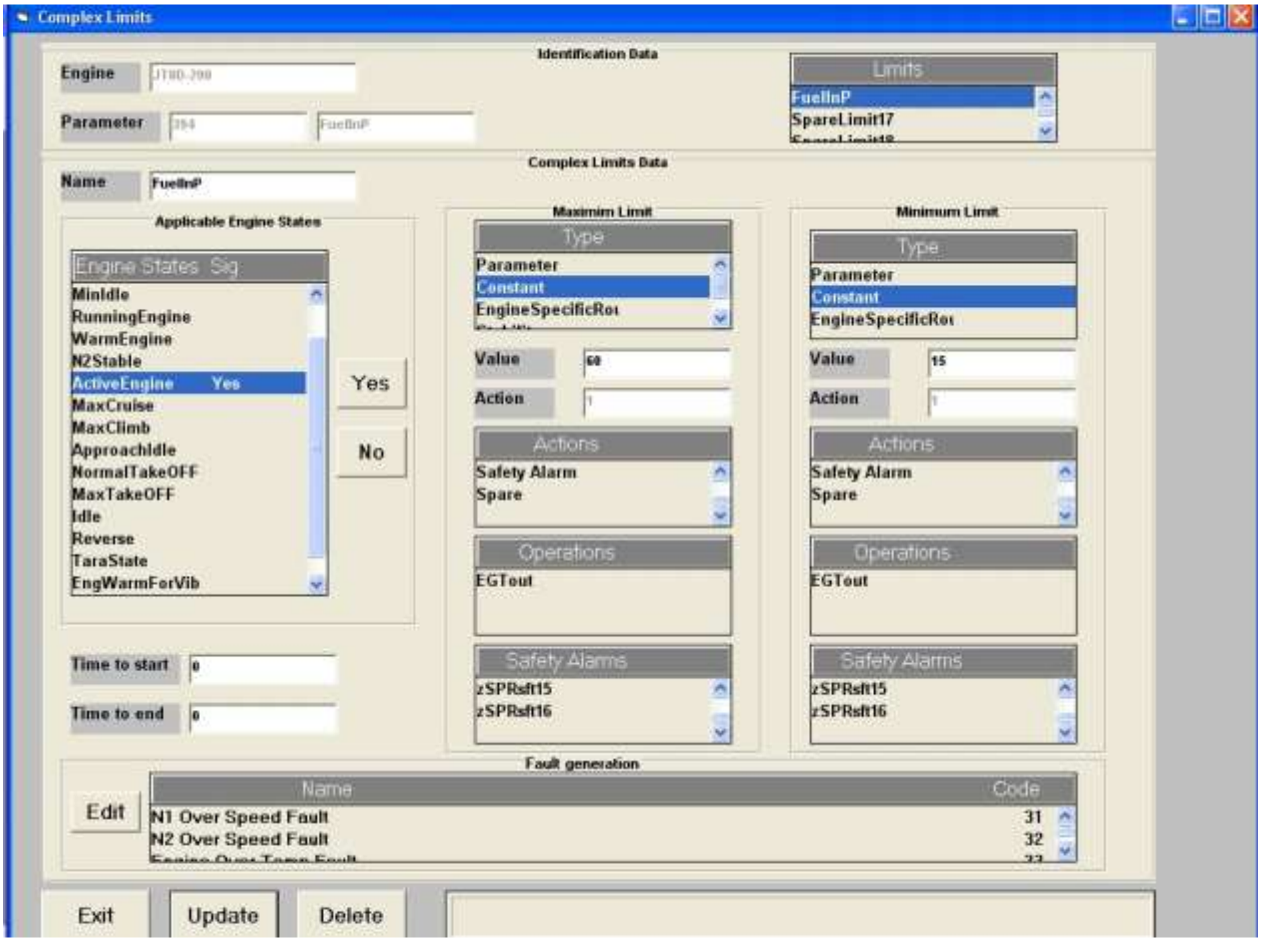



Figure 4.4-B: Complex limit definition screen.

### 4.5. Schedules/Graphic elements

MCTC allows the user to define schedules and other graphical entities. Graphs are defined digitally as an ordered set of points. The following graph types are supported:

- Two dimensional (XY) graph
- Two dimensional band (XYY) graph.
- Three dimensional (XYZ) graph.

MCTC supports linear or quadratic extrapolation of schedule data.

Graphs can be:

- Displayed during engine test run and in the post-run analysis.
- Used to check operational limits of engine
- Used to check engine performance as part of test-procedure tracking.
- Displayed in post-run analysis.
- Used in the formulas of derived parameters.
- Printed.

Example:

Figure 4.5-A displays a band (XYY) graph which defines the N2 limit for idle state.

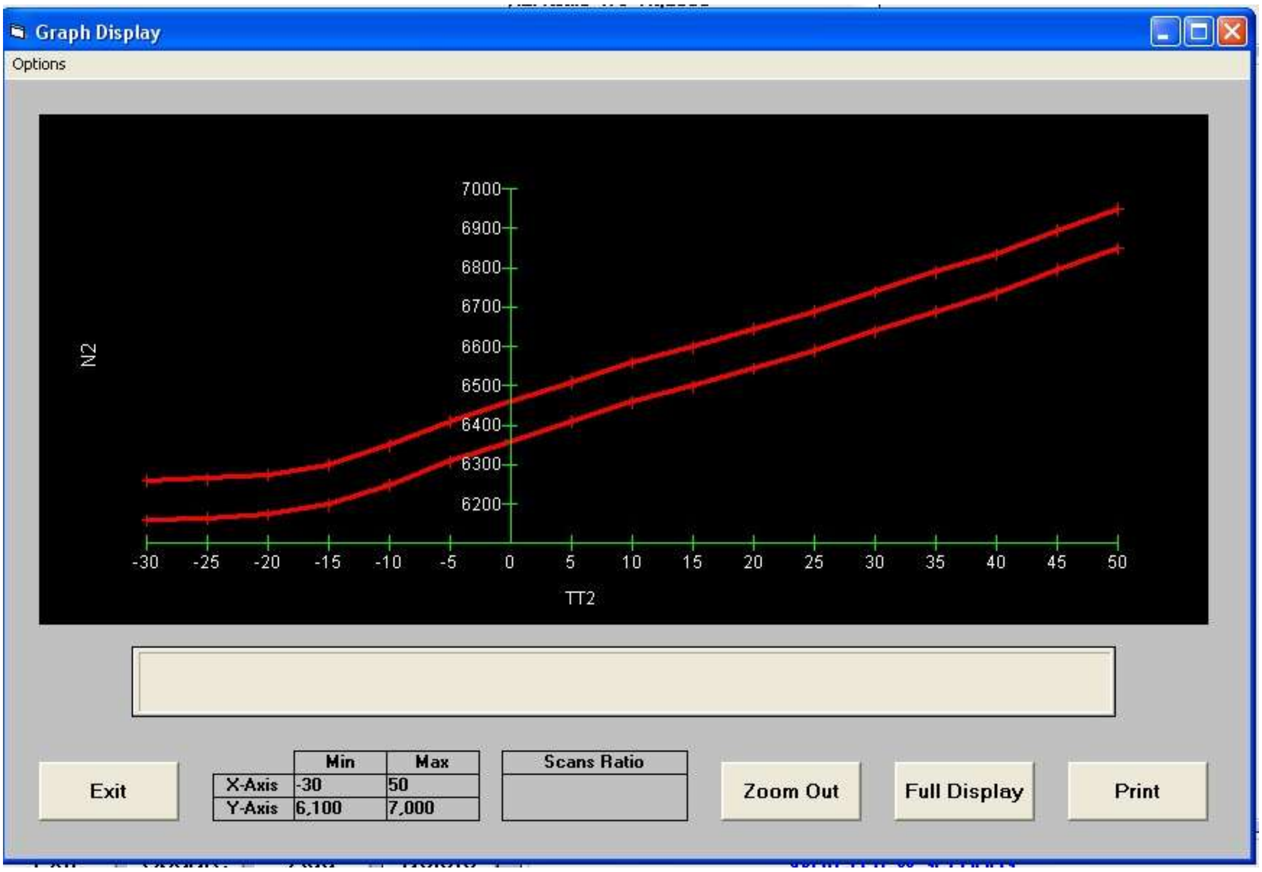



Figure 4.5-A: Band graph display screen.

## 4.6. Parameters' Lists

Parameters can be grouped into an unlimited number of different sets using the Parameters' List definition option. Some lists are required by the system (named "system lists"). Other lists are defined and used by the users - in the post-run analysis reporting functions and to capture specific data items during specific test steps for the formal Run Summary report. These are called "users lists".

<u>Example</u>:

Figure 4.6-A displays the parameters list "Map4List" which defines the parameters that are used for the JT8D-200 map4 analyzing tool.

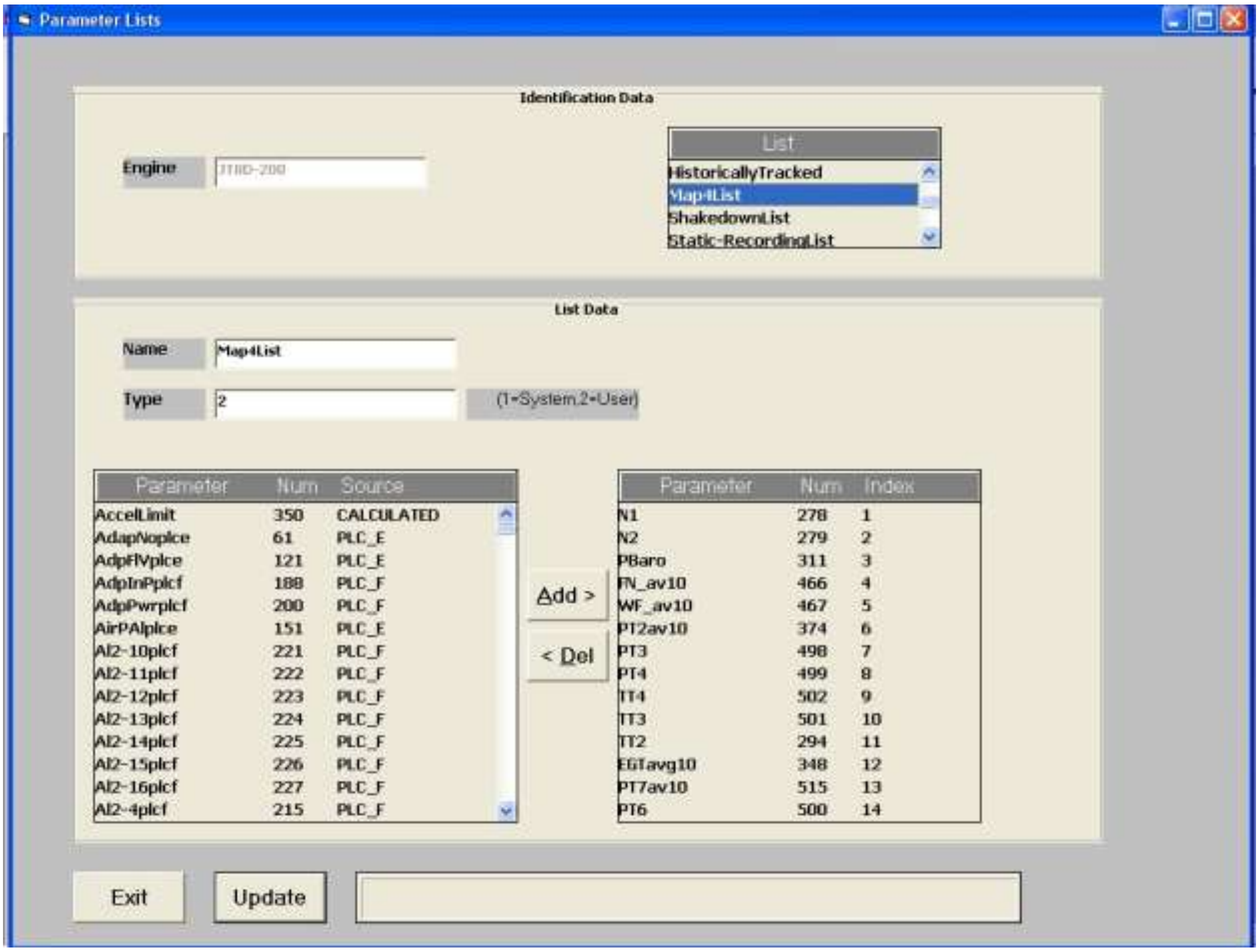



Figure 4.6-A: Parameters list definition screen.

### 4.7. Faults/Warnings

Engine or Test Cell faults as well as "limit-approaching" alarms are identified and treated by MCTC. Appropriate response action(s) can be defined for any such event.

Engine faults are most often identified and reported by external (usually "engine-mounted") devices. Test cell faults will be identified by an external controller (e.g. PLC) or by the limit-violation tracking mechanism.

For each fault/alarm various response actions may be specified. The set of actions is identical to the response actions for limit-violation (see section 4.4 above). Fault detection can be filtered by a time filter (activate response actions only if fault persists for a specified period of time) as well as a "logical" filter implemented by a derived parameter (if the value of the parameter is FALSE - ignore the fault condition).

<u>Example</u>:

Figure 4.7-A describes some of the faults and alarms defined for a specific test cell.

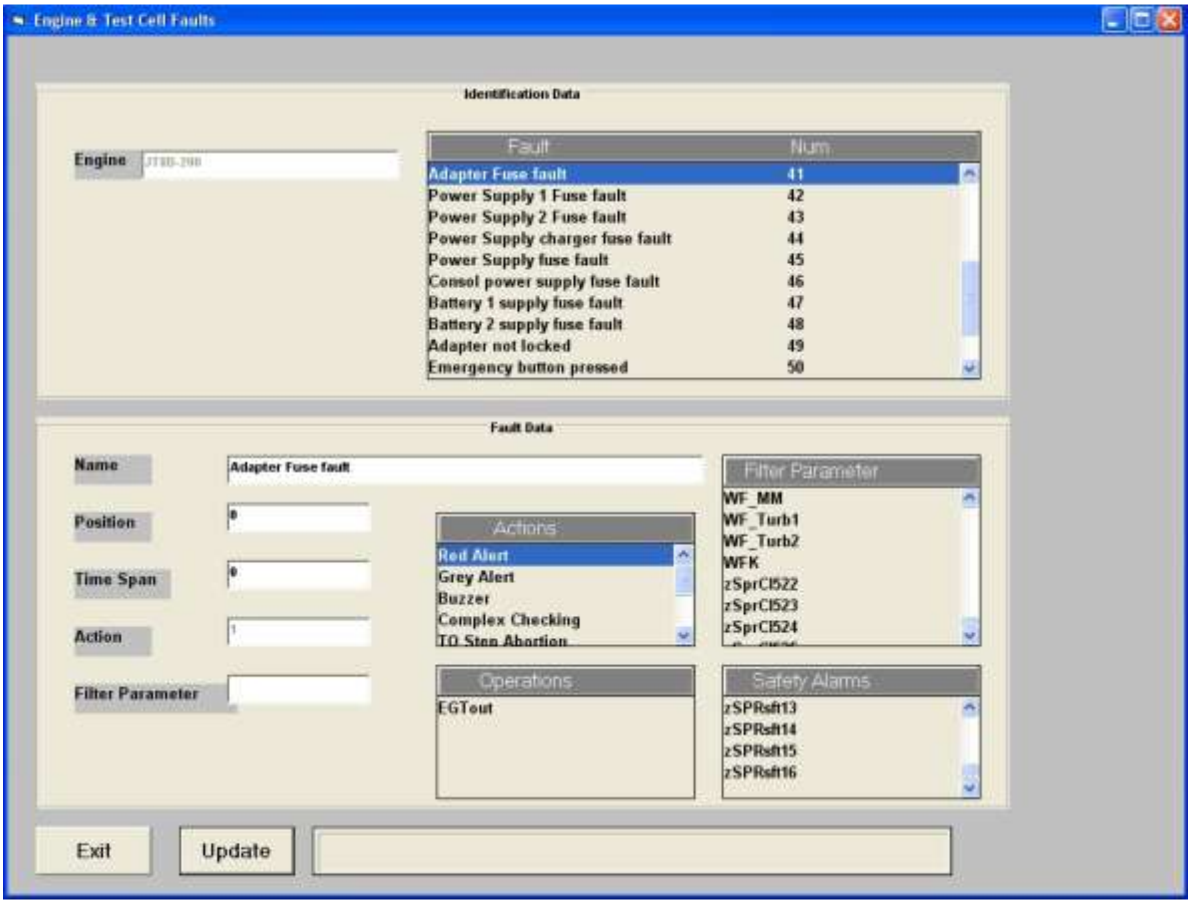



Figure 4.7-A: Fault definition screen.

### 4.8. Control Functions (output operation)

MCTC can control the engine and the test cell facilities using the "output operations". Various levels of control are available - from no control at all ("Monitoring only") to complete control of all engine and Test Cell operation ("Automatic Testing").

Control Functions can be:

- Simple: Set a given value to a single output parameter.
- Complex: controlled update of 1 or 2 output parameter from given initial values to target values, to achieve target values in other input parameters (e.g. - complete feedback closed loop).

Control functions can be activated:

- Automatically as part of a test-procedure definition.
- Automatically as a response to a limit-violation, fault or alarm condition.
- Manually by the user.

### 4.9. Safety Alarms

For critical (dangerous) situations the system response can include a display of a "safety alarm" message. This display which covers large portions of the screen (with a "cannot be missed" red background) must be acknowledged by the operator before normal operation can resume.

## 4.10. Display Screen Layout

The "Screens and Displays" utility allows the user to plan a virtually unlimited number of display screens to be used during the test run and the post-run analysis phases.

Different display screens are defined for the DAC and the MTC.

Any parameter can be included in the display. A multitude of display formats allows the optimization of the displays to the needs of a wide variety of Test Cell environment. Display format include (partial list):

- Digital display (in various sizes).
- Discrete display (On/Off)
- Graphical display (Slide bar, temperature bar etc.).
- Historical display (graph of parameter(s) vs. time).
- Schedule display (graph of parameter as a function of another parameter).

The actual screen that is displayed during engine testing can be:

- Set automatically by the test-procedure definitions.
- Set manually by the user.

Example:

Figure 6.4-B (in section 6.4 below) displays a selection of parameter display formats selected for the CFM56-3 engine Test Cell.

### 4.11. Run reasons / Run types

The reason(s) which necessitated the testing of the engine in the Test Cell (e.g. Parts that were replaced, maintenance operations performed or faults reported during previous engine operation) dictate the type of tests that need to be carried out. MCTC allows the user to define the complete list of "Run Reasons".

Example:

Figure 4.11-A shows a small portion of the run-reasons for the F110-100 engine.

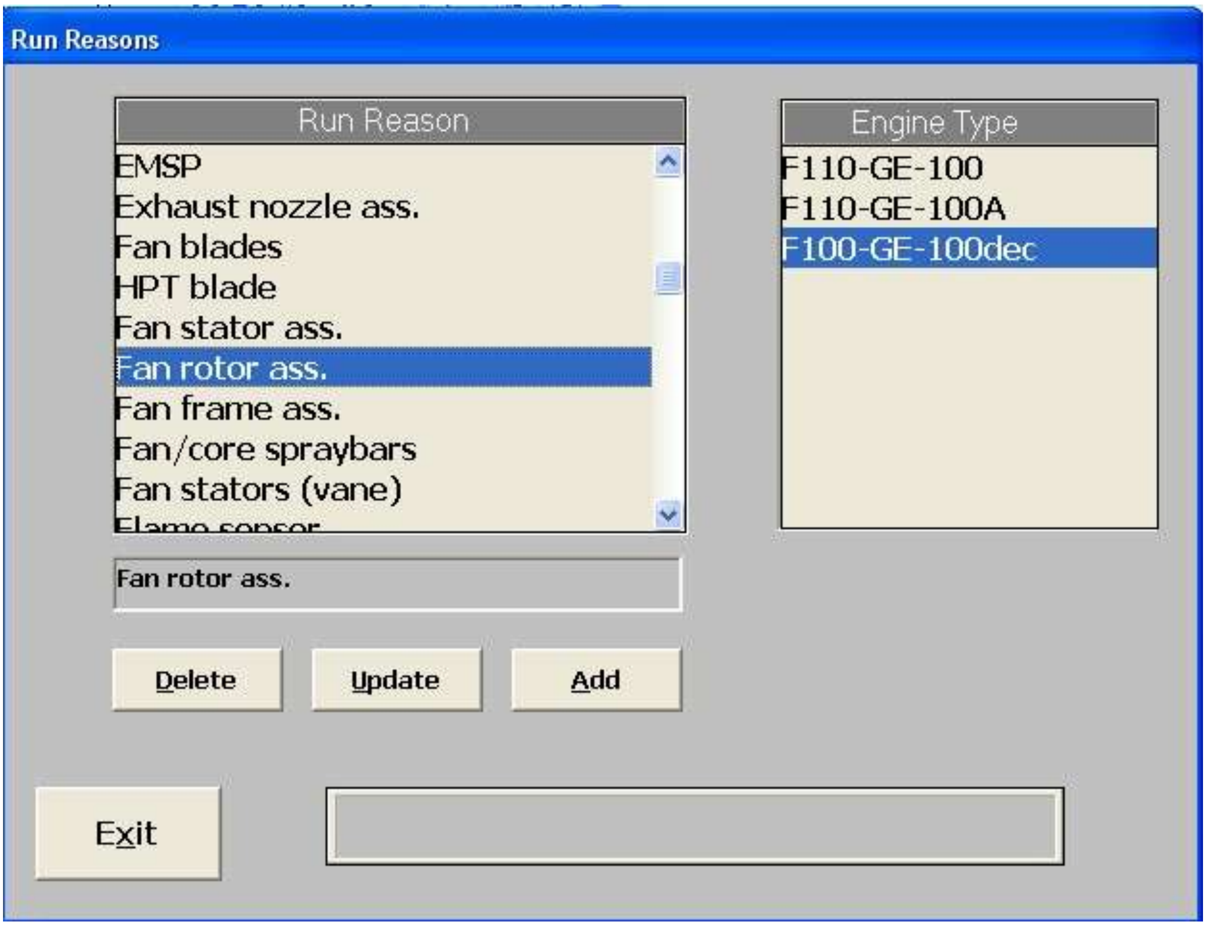



Figure 4.11-A: Run reasons definition screen.

The required test procedures are defined as a set of separate "Run Types" where each run type defines the procedure for a specific test.

Example:

Figure 4.11-B shows the run-types for the F110-100 engine.

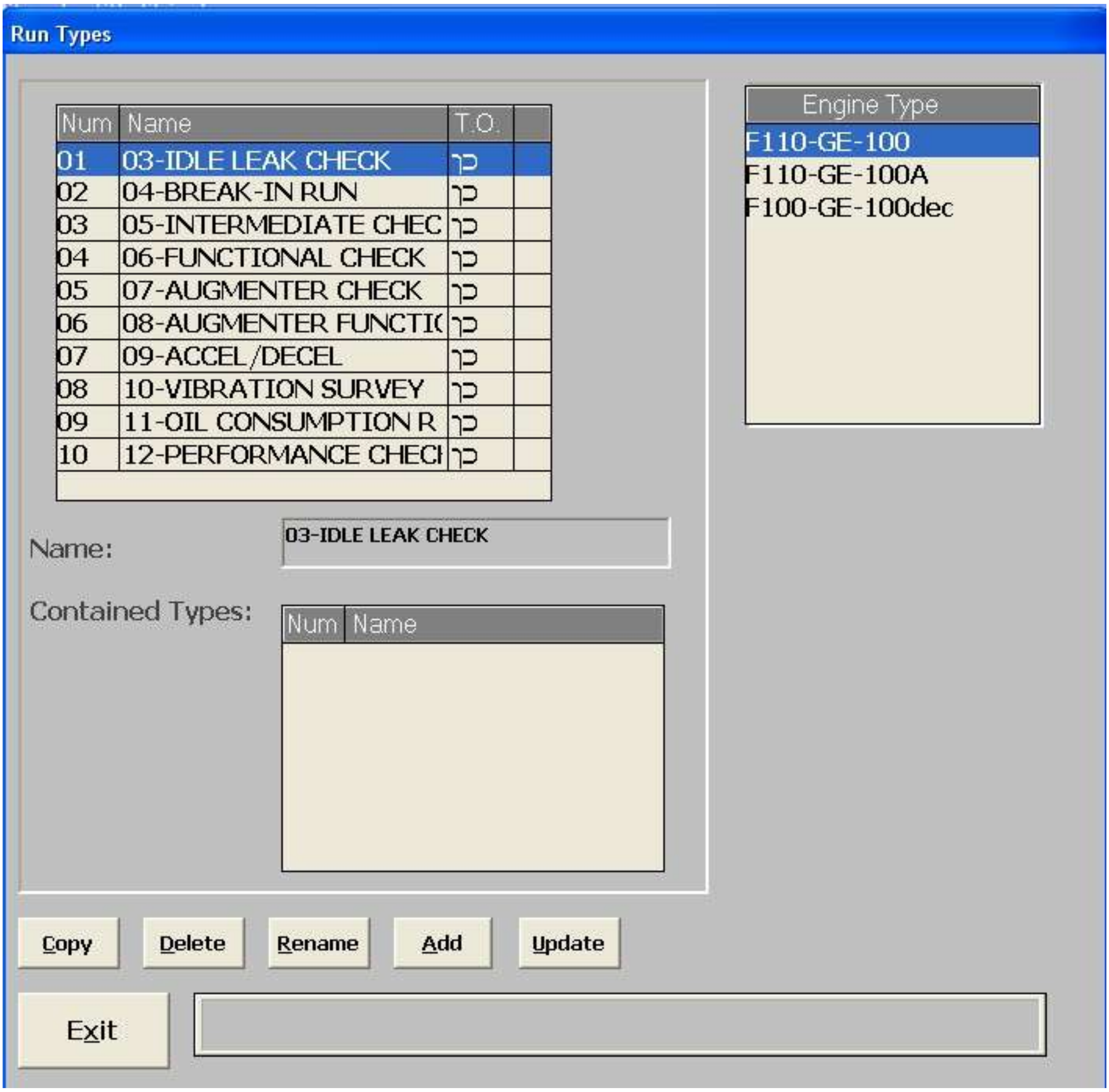
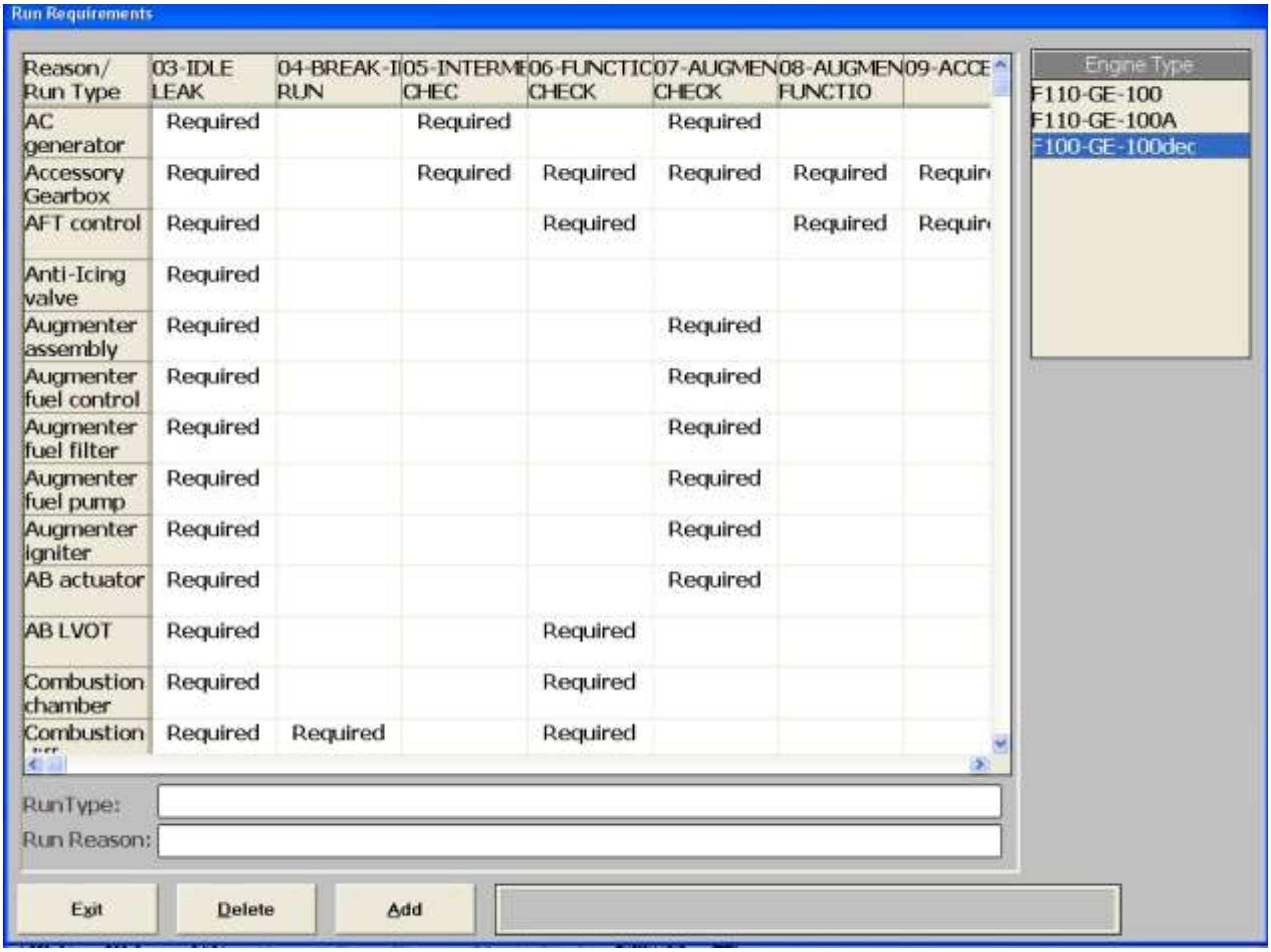



Figure 4.11-B: Run Types definition screen.

Each Run Reason defines a set (0 or more) of tests (run-types) which have to be executed on the engine. MCTC allows the user to define the table ("matrix") that defines which run types are required for each run-reason.

<u>Example</u>:

Figure 4.11-C shows a small portion of the requirements matrix for the F110-100 engine.




| Reason/ Run Type | 03-IDLE LEAK | 04-BREAK-I RUN | 05-INTERME CHEC | 06-FUNCTIC CHECK | 07-AUGMEN CHECK | 08-AUGMEN FUNCTIO | 09-ACCE |
|---|---|---|---|---|---|---|---|
| AC generator | Required | | Required | | Required | | |
| Accessory Gearbox | Required | | Required | Required | Required | Required | Requir |
| AFT control | Required | | | Required | | Required | Requir |
| Anti-Icing valve | Required | | | | | | |
| Augmenter assembly | Required | | | | Required | | |
| Augmenter fuel control | Required | | | | Required | | |
| Augmenter fuel filter | Required | | | | Required | | |
| Augmenter fuel pump | Required | | | | Required | | |
| Augmenter igniter | Required | | | | Required | | |
| AB actuator | Required | | | | Required | | |
| AB LVOT | Required | | | Required | | | |
| Combustion chamber | Required | | | Required | | | |
| Combustion | Required | Required | | Required | | | |

Figure 4.11-C: Requirements matrix definition screen.

35 Hayarkon St., 51204 Bnei-Brak, ISRAEL Tel: 972-3-5780480 Fax: 972-3-5780525
e-mail: info@magnus-eng.com

## 4.12. Test Procedures

The Test-Builder utility is a powerful tool that allows the user to fully "integrate" the required test procedures into the test-run tracking mechanism of MCTC.

Each test procedure is translated into a single test-type.
Each test type is defined as a set of test "steps".

Example:

Figure 4.12-A shows some of the steps which make up the "Acceptance" test of the CFM56-7B engine.

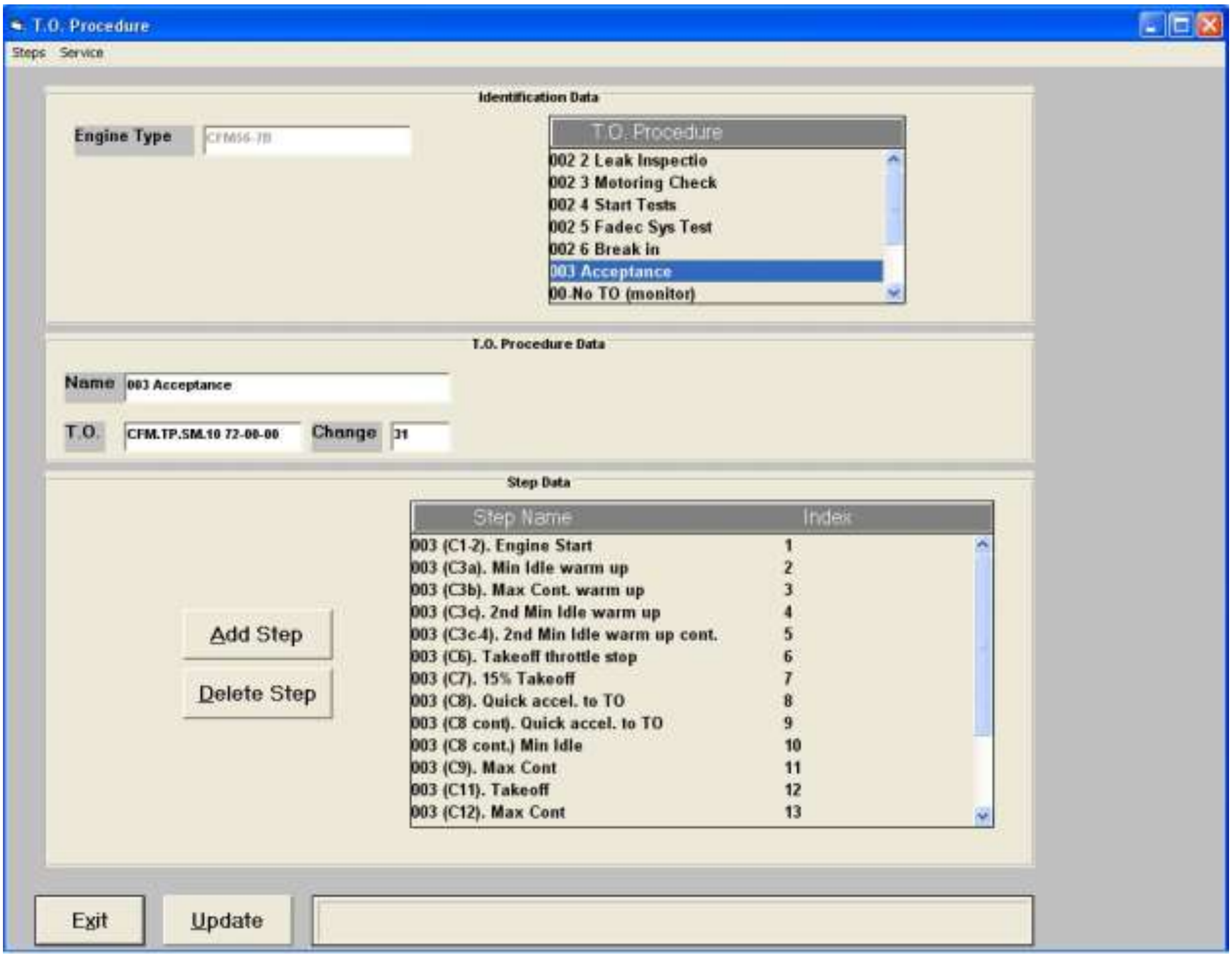



Figure 4.12-A: Test procedure definition (main screen).

Each step is translated into a single control form that will be displayed to the user during the test-procedure execution. Each step is made up of the following components/attributes (partial list):

- Descriptive name
- User instructions (textual operational instructions)
- Display screens identification (in DAC and MTC)
- Required Pre-step stabilization time
- Step duration ("time")
- Step Conditions.

Step conditions are divided into 4 groups:

- Pre-Execution conditions

  This set of conditions must be fulfilled for the "pre-step" stabilization time before actual step conditions are checked. These conditions will usually ensure the engine has been brought to the appropriate conditions for the required test.

- Execution conditions

  This set of conditions must be fulfilled for the duration of the step "time" for the test to be completed. Usually these conditions ensure the engine has stabilizes for a specified period of time (but other cases such as tracking of a specific transient operation are not uncommon).

- Early-Termination conditions

  This set of conditions - if fulfilled for a predefined (short) period of time - will cause the step to be "abnormally" terminated and the system will move on to check the "post-execution" conditions. Early termination does not mean the test has failed - step status is determined only by the Post-Execution conditions.

- Post-Execution conditions

  This set of conditions determines if the engine has passed the required test. For successful completion of the step all the Post-Execution conditions must be fulfilled (successful).

<u>Example</u>:

Figure 4.12-B displays a single step of the "Acceptance" test of the CFM56-7B engine. No "Early-Termination" conditions are defined - therefore in order for this step to be completed N1R parameter value must be stable around the required value for a period of 600 continuous seconds. At the end of the step averaged data values are recorded (sent to the final run report) and the operator is asked to confirm that the test was executed with no fan speed modifiers installed.

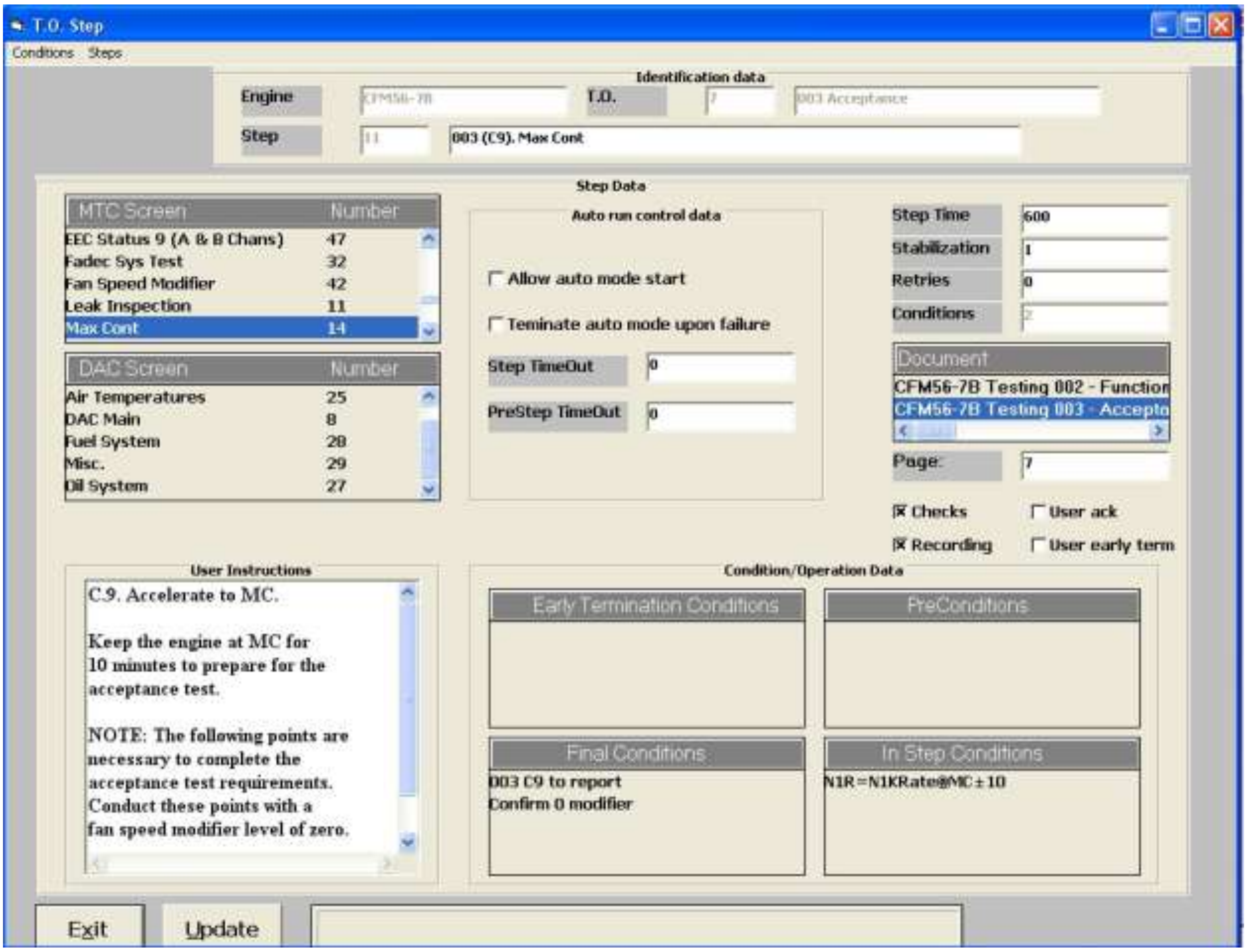



Figure 4.12-B: Test procedure "Step" definition screen

Test "Conditions" can be one of the following types (partial list):

- **Computer**: A condition to be checked by the MCTC automatically. Success status is determined by the computer.
- **Operator**: A manual check (e.g. - "check for oil leaks"). Success status is determined by the operator.
- **Snapshot**: The current value of a specified set of parameters (a parameter list) is sent to the printer and a logical event is recorded (for later extraction in various report utilities). Status is always "successful".
- **Operation**: An output (control) operation is initiated. Status is always "successful".
- **SYSPARM Update**: Generation of a system parameter value.
- **Applicability**: Must be a "pre-execution" condition. If this condition is not successful - step need not be executed.

Example:

Figure 4.12-C displays the internal definition of a "computer"-type condition.

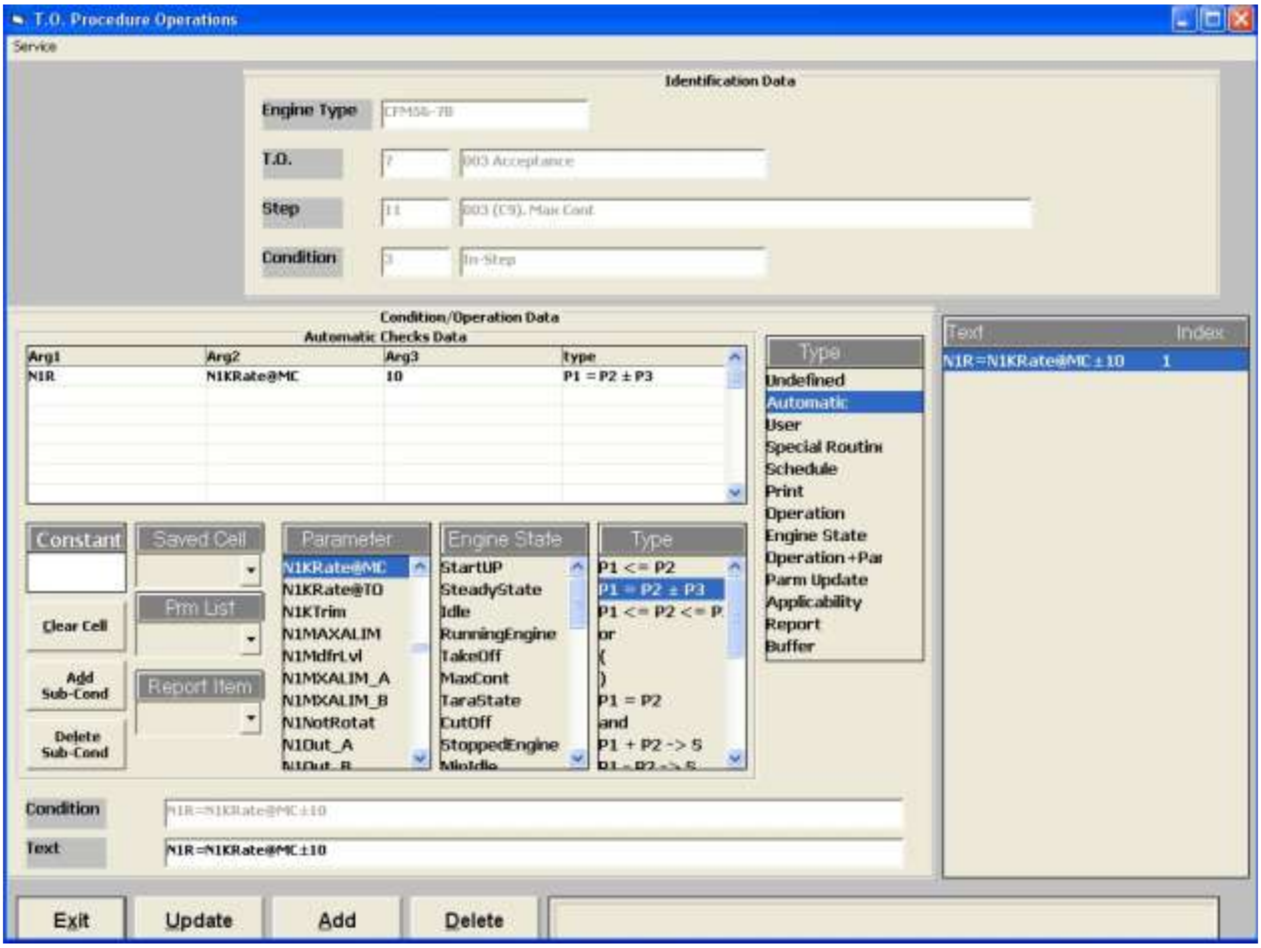



Figure 4.12-C: Test procedure "Condition" definition screen

# 5. Engine testing procedures

## 5.1. Test Run initiation

A test run (a new engine or another test for the current engine) is initiated via the test run "Entry Screen".

In order to initiate a **new test run** the operator has to provide the following information:

- Engine type.
- Engine serial number.
- Additional work identification data (which is usually "Test Cell specific" and is set up to meet specific user's requirements).
- Run reason(s) (optional).
- Run type (required).
- Operator identification number ("tester 1").
- Supervisor identification number ("tester 2").
- Dynamic test properties.

Notes:

- According to the selected run reasons, the required run types are automatically displayed.

- One run reason which can always be selected is "general". Selecting this reason will cause the system to display **all** run types defined for the selected engine type.

- Run types that are not **required** by any run reason (other than "general") are marked as "optional".

- One (and only one) run type must be selected for each run-continuation. The "No T.O. (Monitoring)" run type is always available (regardless of selected run reasons).

- Dynamic properties include features like scan rates (Normal/Transient/Static Recording), Filtering algorithm definition etc. These can only be modified by a properly authorized user.

When initiating a **run-continuation** the operator has to provide the following information:

- Update list of run reasons (optional)
- Select one run type (required)
- Update Dynamic run properties (via "Service" function - optional).
- Update operator(s) identification data (optional).

The list of run types is automatically maintained to indicate the status of each test type (Required, Failed, Successful, Optional).

Example:

Figure 5.1-A displays the test run entry screen set up to perform CFM56-7B engine run.

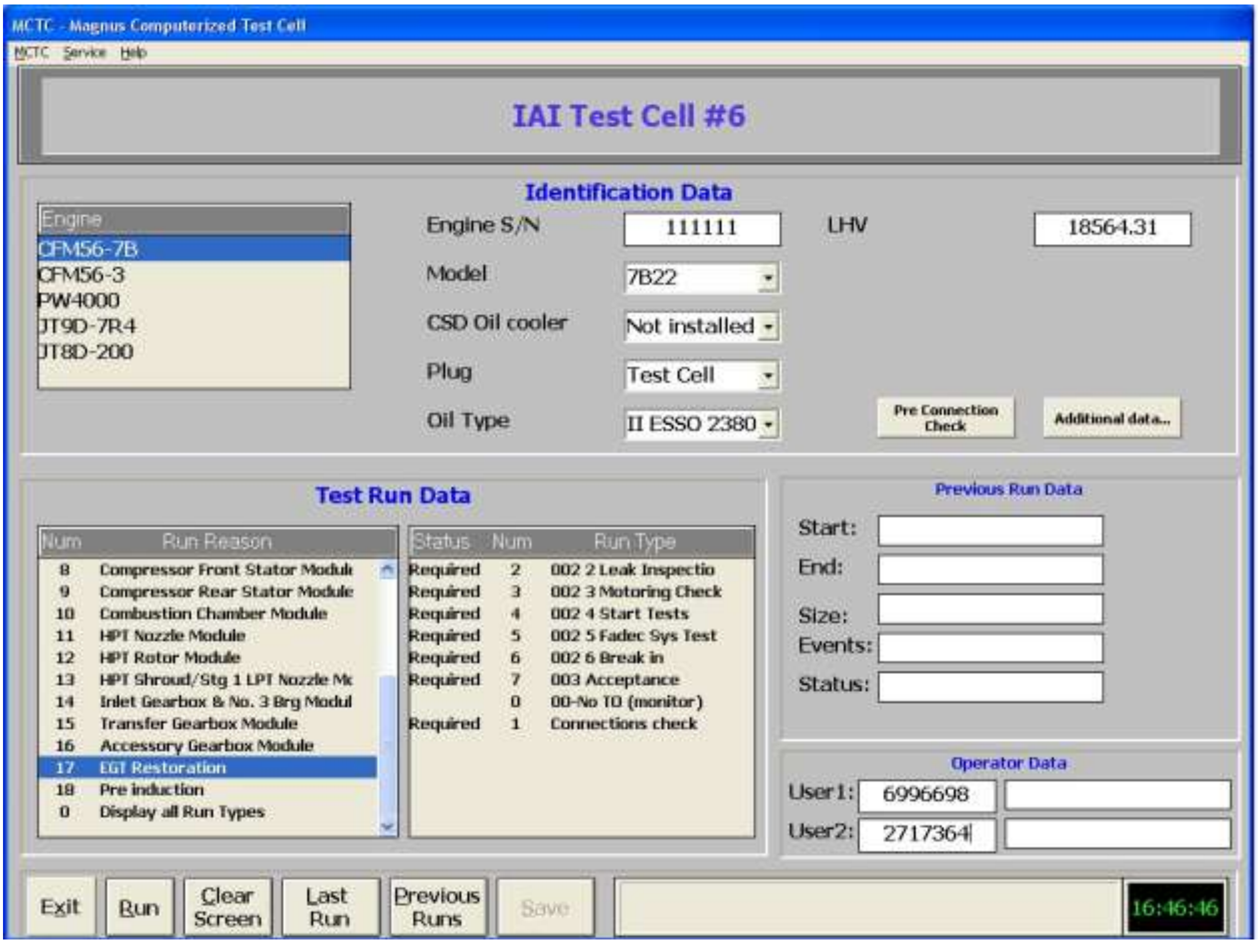


Figure 5.1-A: Run initiation screen.

35 Hayarkon St., 51204 Bnei-Brak, ISRAEL Tel: 972-3-5780480 Fax: 972-3-5780525
e-mail: info@magnus-eng.com

## 5.2. Test Run execution

During actual engine testing the DAC and the MTC cooperate in the tasks required to monitor and control the test run.

DAC functions include (partial list):

- Data acquisition from all external sources.
- Data filtering.
- Data storage (to hard disk).
- Critical parameter display.

  The DAC screen usually displays a fixed set of critical parameters (but, as explained before, any number of displays may be defined and shown).

- Limit Violation monitoring and treatment.

  For each detected Limit violation an appropriate message is displayed on the DAC screen in RED color. Any specified response actions are carried out by the DAC and/or MTC. .When the limit condition no longer exists the display is colored GREEN.

- Faults Conditions monitoring and treatment.

  For each detected Fault or alarm condition an appropriate message is displayed on the DAC screen in RED color. Any specified response actions are carried out by the DAC and/or MTC. When the fault/alarm condition ceases to exist the display is colored GREEN.

- If the response action to a limit violation or a fault is "safety alarm" - the appropriate "safety alarm" message is displayed on the DAC screen.

- Required output operations are carried out by the DAC by delivering appropriate commands/values to external subsystems (e.g. PLC).

MTC functions include (partial list):

- Test-related parameter display. The MTC display screens are designed to match specific test steps. The display is updated automatically to show the parameters that are relevant for the current test step.
- Handle user requests for various recording operations (change from Normal to Transient recording, perform Static recording, record a manual "snapshot").
- Accept values entered by the user for "user-input" parameters.
- Allow user to dynamically modify the current MTC display by adding/deleting selected parameters.
- Test procedure tracking:

  - Automatic tracking of the current test procedure.
  - Automatic MTC (and possibly DAC) display update.
  - Display of the text (operator instructions) connected to the current step.
  - Dynamic display of the status of the conditions that are currently being checked.
  - Automatic timing of pre-step and step duration.
  - Automatic execution of output operations as defined in the Test procedure.
  - Automatic "snapshot" recording.
  - Generation of values for "system parameters".
  - Allow manual control of test sequence (user can "stop" the current step and move to any other step).
  - Accept user input to determine the status of any "operator" conditions.

- Event recording: The MTC records (to disk) all the events that happen during the run. This includes limit violations, faults, alarms, external subsystem problems, and all the test-procedure events (successful conditions, failed conditions, aborted steps etc.). These records serve as the basis for post-run reports and analysis functions.

<u>Example</u>:

Figure 5.2-A displays an MTC rest run screen for the CFM56-3 engine. The test-procedure form shows the startup step of the functional test. The system is currently waiting for the in-step conditions to be fulfilled (Minimum-Idle conditions have not been achieved, and the startup process has not been completed yet).

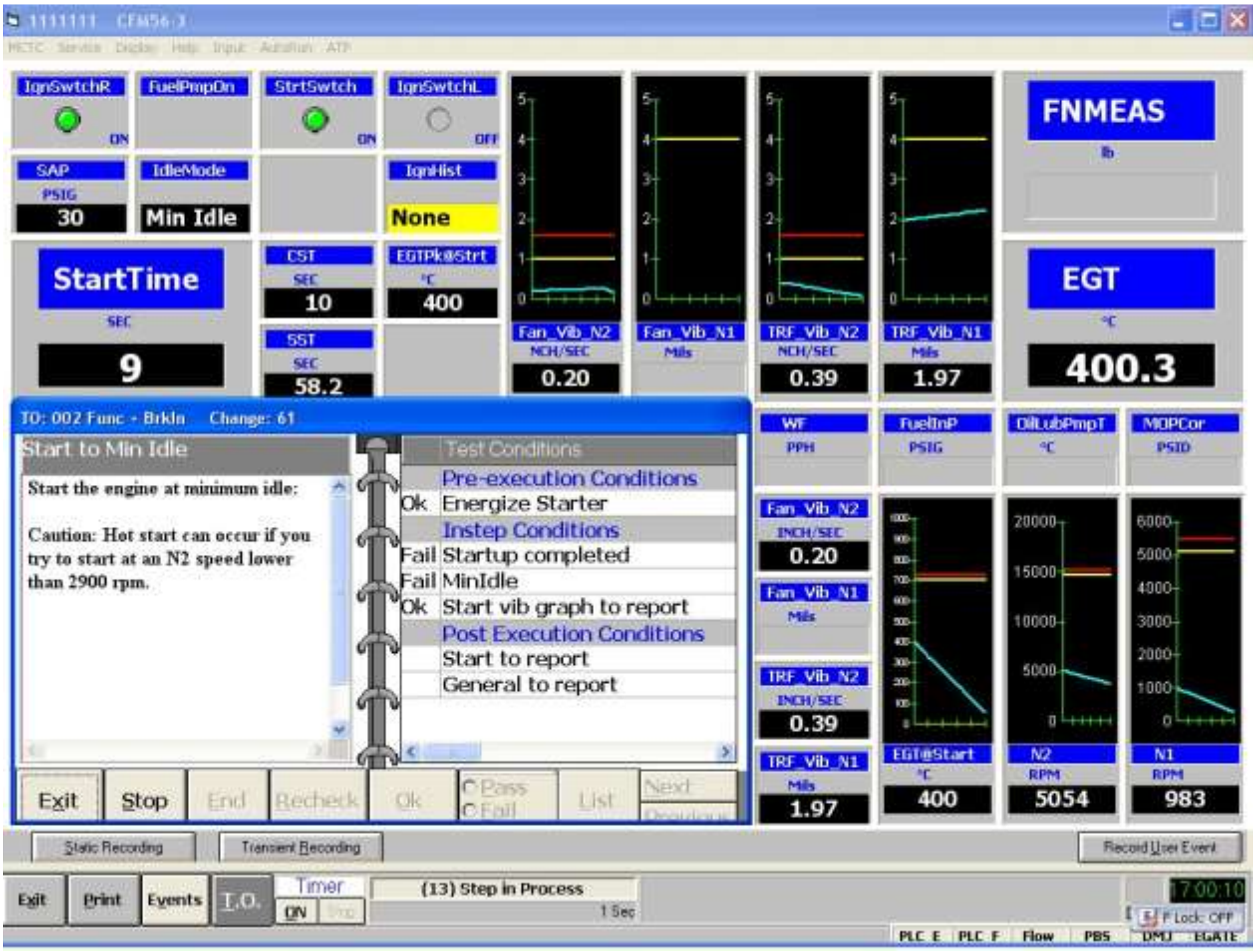



Figure 5.2-A: Engine test run screen (MTC) with open T.O. "book".

### 5.3. Test Run termination

Engine test run is always terminated by the operator (the test procedure may have been completed, or may have been prematurely aborted). The run summary screen allows the user to define the status of the last run-continuation, as well as the status of the entire engine testing process. At the end of each run-continuation the user defines:

- Status of last run-continuation (Passed / Failed) (required).
- Status of engine-testing (Passed / Failed) (optional).
- Free text with operator comments (optional).

**Note**: The determination of the test status (Passed/Failed) is **ALWAYS** the responsibility of the operator:

- External signs which are not available to the computer (e.g. noise, smoke) may indicate that the engine is not fully operational, even if all the defined (computerized) checks have been successfully completed.
- Some required tests may not be executable (or may fail) due to external conditions which are not related to the engine being tested (e.g. test cell facility malfunctions) - while the engine is fully operational.

However, the concise run summary report will always indicate any required tests that have not been successfully completed, any limit violations and any engine faults which have occurred during the testing. These should be taken into account when the operator decides on the run status.

In order to complete the engine testing the operator should:

- Indicate final Run status.
- Save the run (this will extract the data stored on the DAC and save it in the run archive of the MTC).
- Produce run summary report (optional - this report may always be regenerated using the post-run analysis functions).

Example:

Figure 5.3-A displays the "Run Summary Screen" for a CFM56-3 engine which has just finished a "Functional + Break in" test. The operator has indicated that this specific test was successful, and that all the required testing of the engine has been successfully completed.

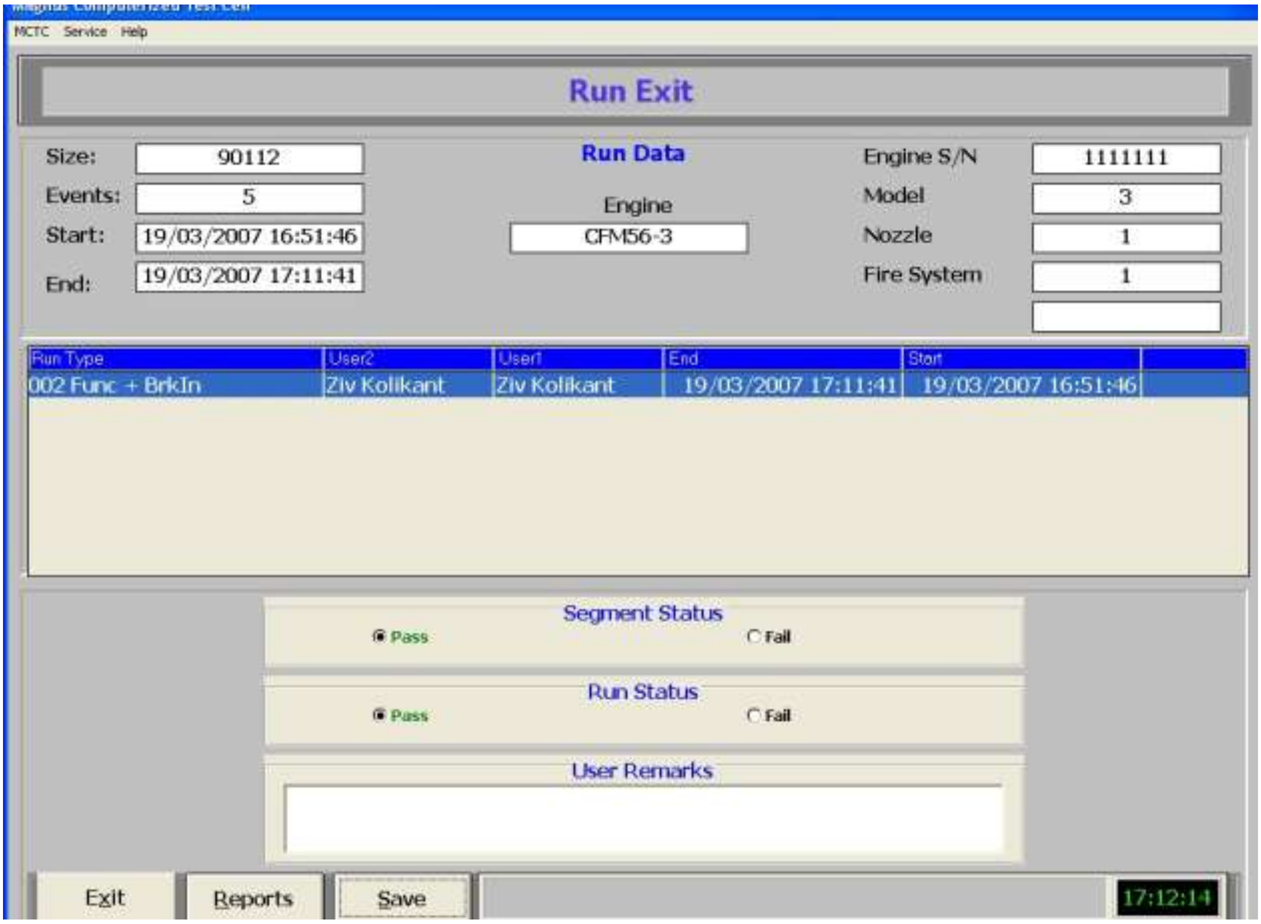



Figure 5.3-A: Engine run-termination (summary) screen.

# 6. Post-run analysis functions

MCTC archives all the information collected during engine test runs. This includes:

- Identification data.
- Raw input data as acquired during the run (e.g. 4 "scans" / second for the entire testing period).
- Events data (limit violations, faults, alarms, test-procedure execution data etc.).
- Operator's comments and status indications.

The post-run analysis utility ("run analysis") allows full and easy access to this data archive in support of in-depth analysis of specific engine tests as well as cross-runs comparisons and long term trending analysis.

Note that archived data may be moved from on-line storage (in MTC) to off-line storage and back using MCTC "utilities" function. This allows investigation and management of test run data in a remote location outside the Test Cell (e.g. - headquarters).

## 6.1. Archive Access facility

The entry screen into the post-run analysis utility is the "Saved Runs List" screen. The user selects the domain of interest by specifying:

- Engine type (required).
- Range of serial numbers (optional).
- Range of dates (optional).

Further selection may be achieved by specifying:

- Run types (e.g. - only runs which include "Acceptance" tests will be selected).
- Limits or Faults (e.g. - Only runs in which a specific limit was violated will be selected).
- Event types (e.g. only runs in which vibration-related events were recorded will be selected).

Example:

Figure 6.1-A displays JT8D-200 tests currently archived. At this stage (as displayed in figure 6.1-A) analysis of all the selected runs may be initiated by the "Multiple Run Analysis" button. By clicking on a single run from the displayed list, in-depth analysis of this specific run may be initiated by the "Single Run Analysis" button.

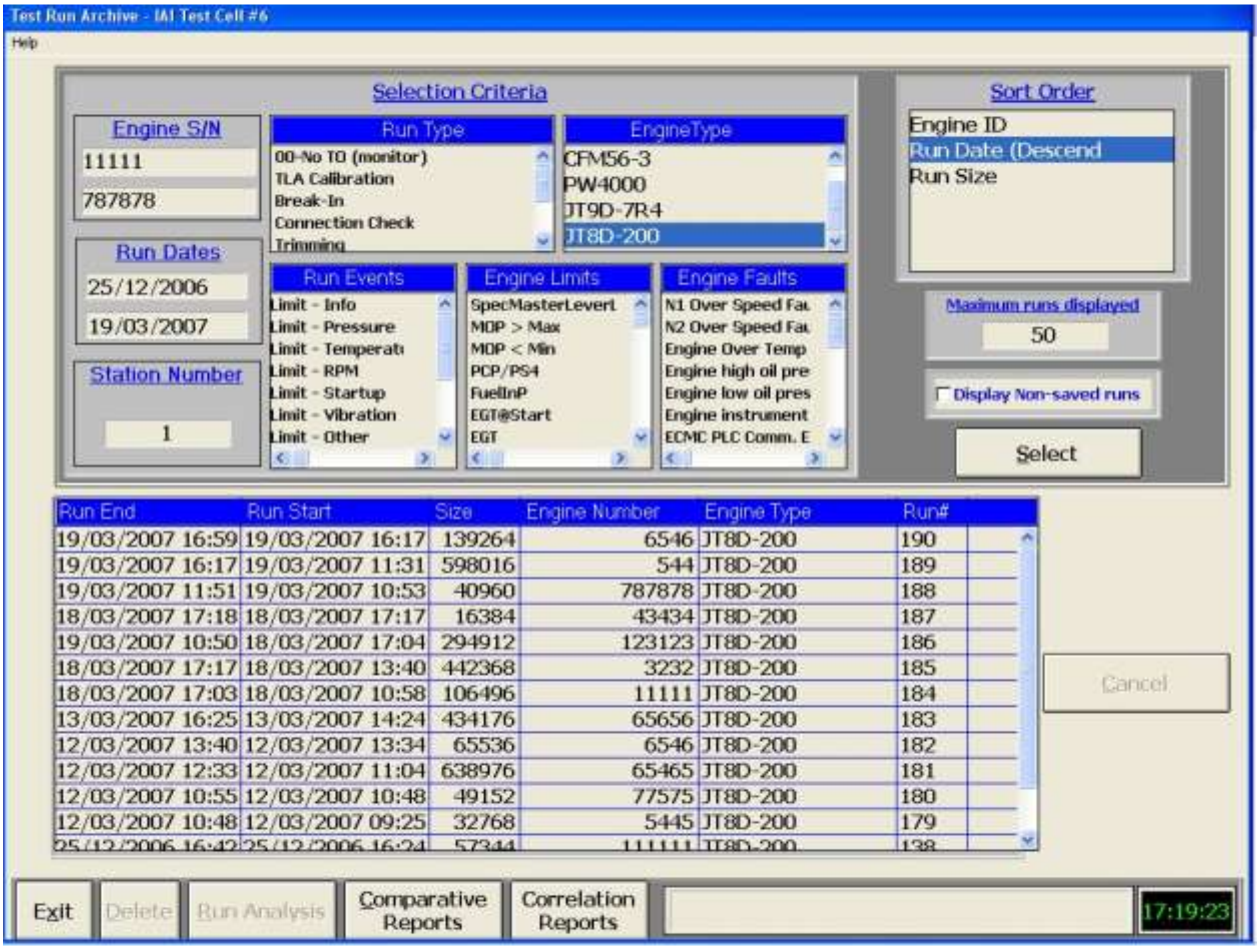



Figure 6.1-A: Post run analysis utility entry screen.

## 6.2. Single Run reports

A variety of reports are available to support both routine test cell operation (e.g. - produce a "Run Form" at the end of each test and attach it to engine's documentation) and in depth analysis of the run.

The "Run Form" report is the formatted summary report delivered with the engine at the end of the run. This report has several customization options:

- Selection from several predefined report templates or creation of new report templates.
- Selection of recorded static and transient points from different recording times.

The range of some of the reports can be modified to include only parts of the complete run by the following user defined attributes:

- Selection of specific run continuations or statement of exact time-range to be investigated. Only scans that fall inside selected time region will be included in the report/graph.
- Specification of a minimum and a maximum value for a selected filter parameter. Note that this can be any parameter (including derived parameters).

For some reports (Snapshot report, Scans Report) the content is defined by selecting a predefined parameter list or by dynamically defining any other (temporary) parameters' list.

The test procedure reports ("Summary Report-Full" and "T.O. Report") will include specific data acquired during the run which affected the Pass/Fail determination of the test conditions. For example - for any failed condition - the reports display both the required (limiting) value as well as the actual value which caused the failure.

The data extracted by some of the reports (e.g. Snapshot report, Scans Report) can be saved as ASCI formatted file or an Excel spreadsheet. This data can then be imported into other (standard) analysis tools for further processing.

Example:

Figure 6.2-A displays the report definition screen for a specific CFM56-3 engine test which included many test runs.

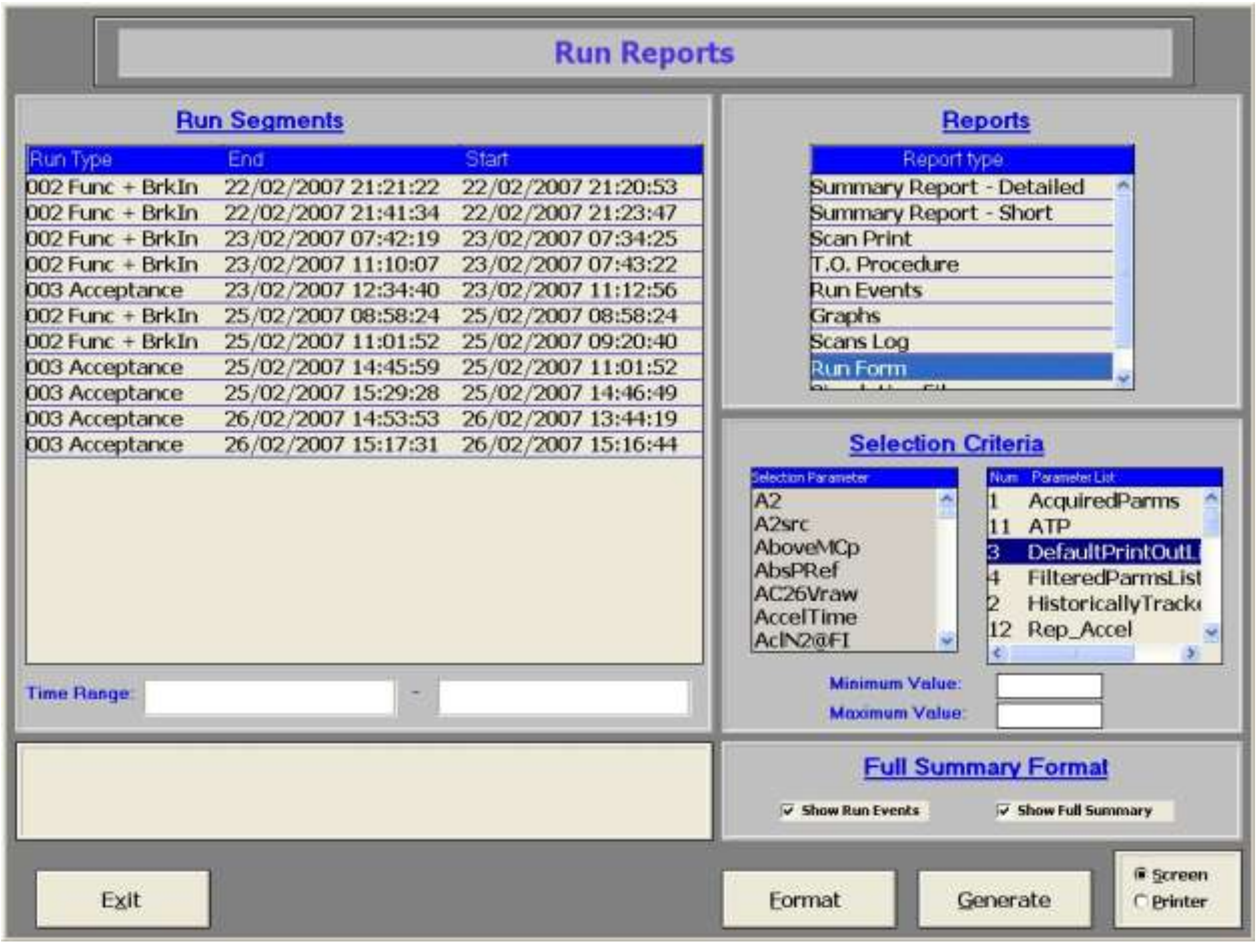



Figure 6.2-A: Report definition screen for a single run

Figure 6.2-A1 displays the first page of the run form report generated for a specific CFM56-3 engine test.

ISRAEL AEROSPACE INDUSTRIES LTD

**_CFM56-3 ENGINE INSPECTION & TEST CERTIFICATE_**

| | | | |
|---|---|---|---|
| Date: | 19/3/2007 | Manual: | CFMI-TP-SM.5 72-00-00 |
| Run End Date: | 26/2/2007 | Manual Revision: | 61 |
| Customer: | [illegible] | Operator | [illegible] |
| Engine Type: | 3C1 | Inspector: | [illegible] |
| Engine S/N: | [illegible] | W.O# | 5232145 |

Standard / Hot Day Performance

| | Standard Day | | | | | | Hot Day | |
|---|---|---|---|---|---|---|---|---|
| | N1K Trgt (RPM) | FNK3 (lbs) | EGTK3 (°C) | N2 K3 (RPM) | WFK3 (PPH) | SFCK3 (lbs/hr/lb) | EGTHD (°C) | N2HD (RPM) |
| Takeoff | 4942 | 23583 | 825 | 14302 | 9456 | 0.401 | 877 | 14632 |
| Max. Cont. | 4804 | 22058 | 797 | 14142 | 8687 | 0.393 | 830 | 14361 |

| | | | |
|---|---|---|---|
| Total Running Time: | | Fuel Type: | JET-A1 |
| Maximum Accel Time: | 7.0 Sec | Oil Consumption: | |
| Observed Accel Time: | 6.5 Sec | | |

Certification Of Inspection

**I hereby certify that the Repair, Inspection, Overhaul and Test of the above mentioned engine has been carried out in accordance with Joint Aviation Authorities Requirements/ Federal Aviation Administration regulations and is approved for return to service. Pertinent details of the repair are on file No. at this agency.**

**EASA Approval No.** [illegible]
**FAA Repair station No:** [illegible]

ENGINE ACCEPTED
Signed: [illegible]
Date: [illegible]

[illegible]

CFM56-3 Summary Log — Page 1 of 4

Figure 6.2-A1: Report layout

## 6.3. Graph generation

The graph generation utility (activated from the report definition screen displayed in figure 6.2-A) can dynamically create various graphs (Parameter vs. parameter, Parameter(s) vs. time) and lets the user investigate these graphs. Friendly Zoom-In and Zoom-Out features allow fast and easy investigation of problematic areas.

Predefined schedules can be plotted and used as the baseline over which actual run data is overlaid.

The graph print out utility generates accurate color graphs (if a color printer is connected) which will clearly and accurately summarize investigative efforts. (This tool is especially useful during data acquisition system installation phase when the stability, accuracy and consistency of acquired parameters have to be verified).

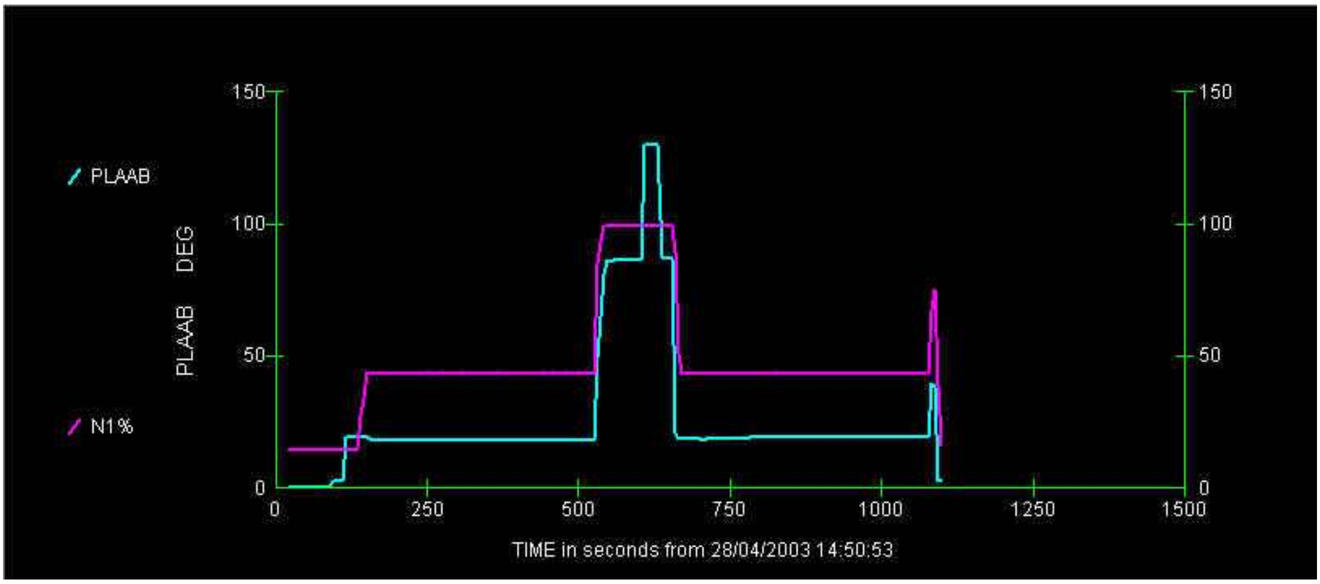



Figure 6.3-A: Two parameters vs. time graph

## 6.4. Playback utility (the Analyzer)

The playback utility is capable of recreating accurately the entire test run and displaying the archived data in exactly the same manner it was displayed during the actual run. Easy and fast access to sections of interest in the run (various events) facilitates investigative efforts following engine or test cell problems encountered during the test run.

The display of the analyzer utility is based on the display screens defined for the MTC. Any defined screen can be selected. Additionally, the user may dynamically update the display by adding or removing any parameter.

In addition to the parameter display, the following displays may be added or removed from the screen dynamically:

- Complete list of run events.
- Complete list of T.O. (test procedure) events (successful/Failed/Aborted steps).
- Specific test procedure "book".
- Complete list of current Engine States values.
- Complete list of current value of internal parameters (used by the T.O. tracking).

Playback operation begins by normal recreation of the original run (i.e. - scans data is displayed on the screen at the original actual rate). The following operations may change the display sequence:

- Stop (halt the display on current scan).
- Play (restart normal playback from current location).
- Fast forward (advance display at a fast rate).
- Fast Rewind (move backwards in time in a fast rate).
- Next Scan (following a "Stop" - advance to next scan).
- Previous Scan (following a "Stop" - retreat to previous scan).
- Click on "event" from events list (start display from the scan which generated the event).
- Click on "event" from T.O. events list (start display from the scan which generated the event).

<u>Example</u>:

Figure 6.4-A shows a replay of a CFM56-3 engine with a list of events recorded during the run. The user has selected a specific event ("Start Time: 55.9 Sec") and the display is located on the event time.

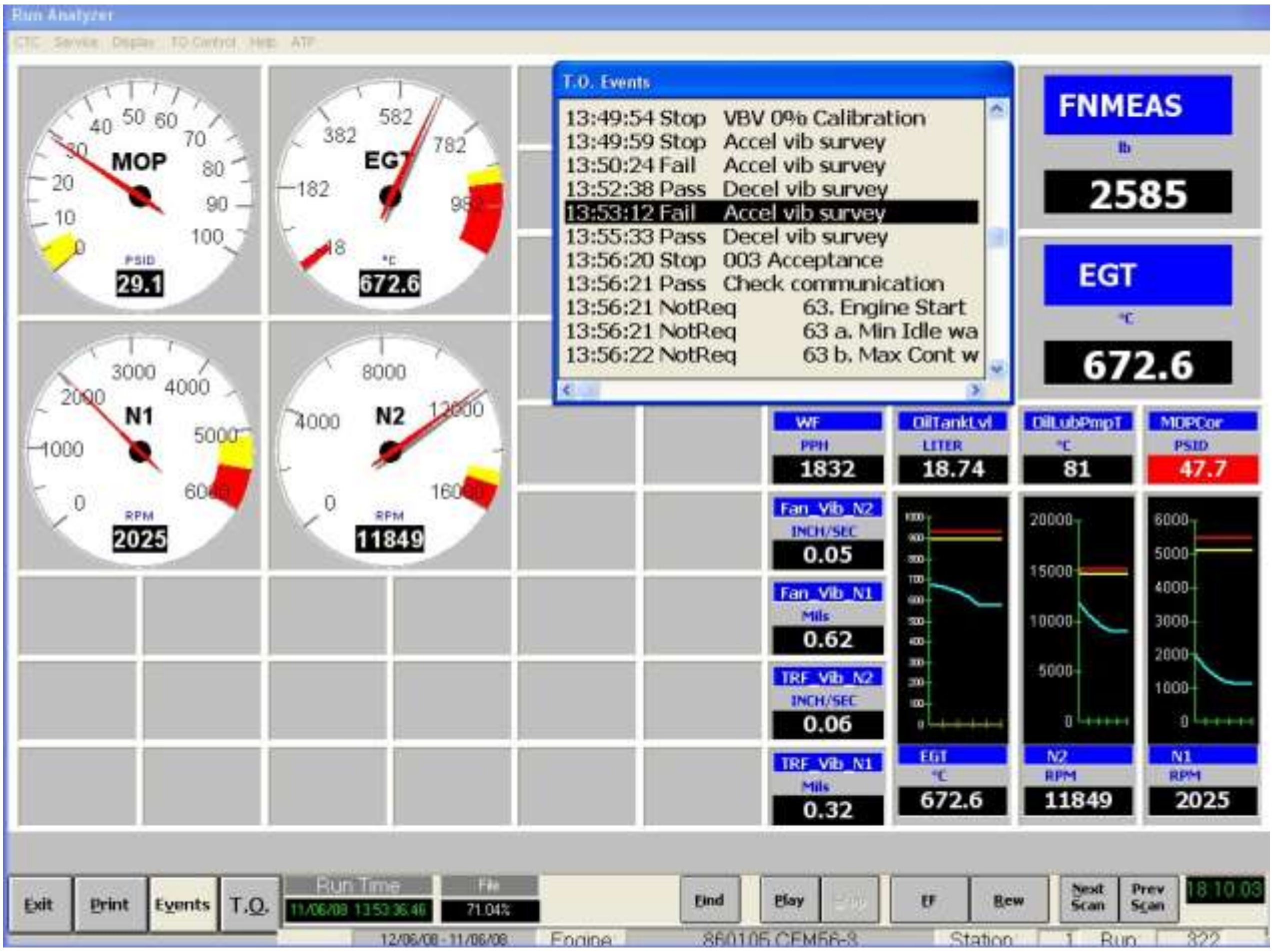



Figure 6.4-A: Playback display with list of events.

Example:

Figure 6.4-B shows a replay of a CFM56-3 engine run with a recreation of a "Acceptance" test. The test is currently being executed.

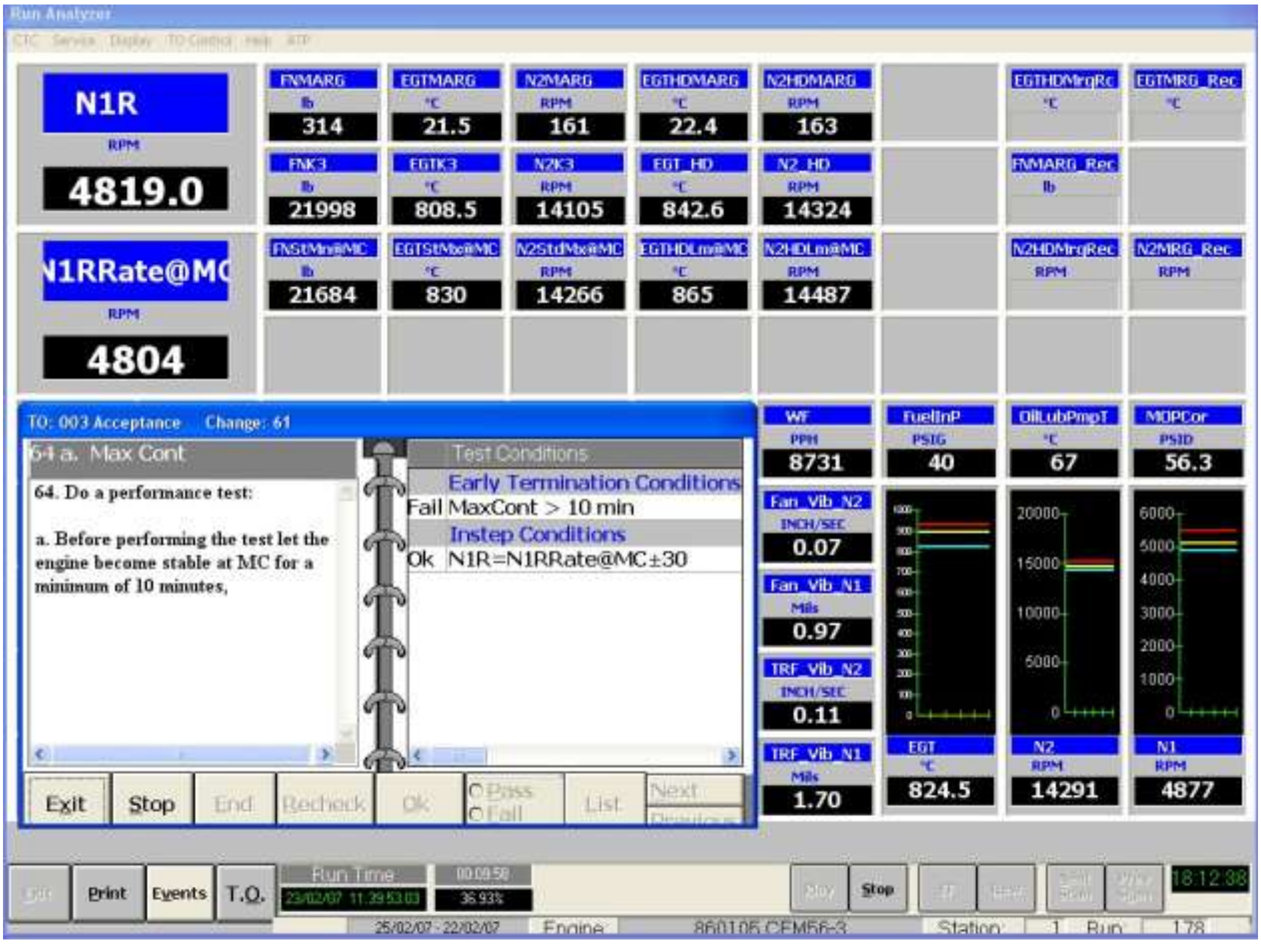



Figure 6.4-B: Playback display recreating a specific test procedure.