

# MTS3 Training

# Environment Setup

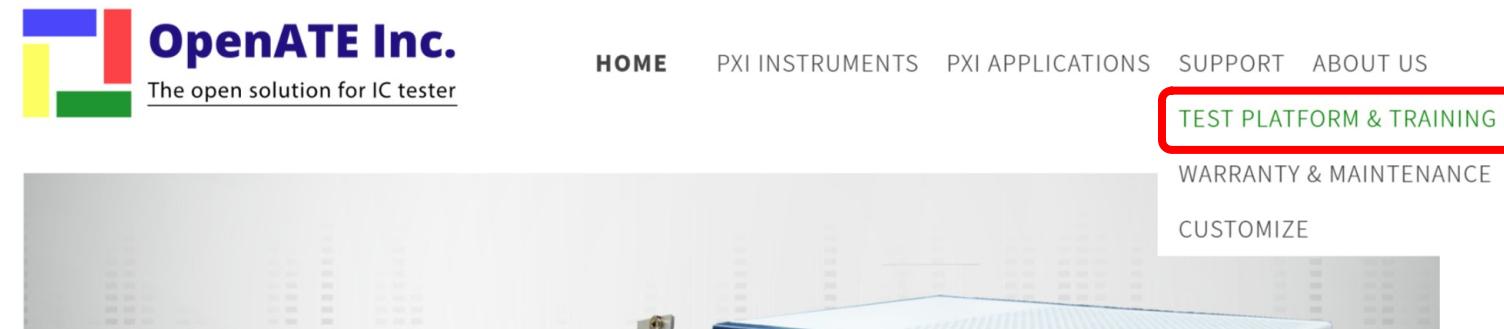
# Preparing for using MTS3

- User should prepare the following equipment:
  - PE card
  - PXI chassis
  - C/C++ program ( Visual C++, Version:2010~2013 )
  - Adobe Reader V11.0
  - PC or NB
  - 68 pin vhdcI cable
  - DUT board

Visual C++  
Version:2013 (Standard)

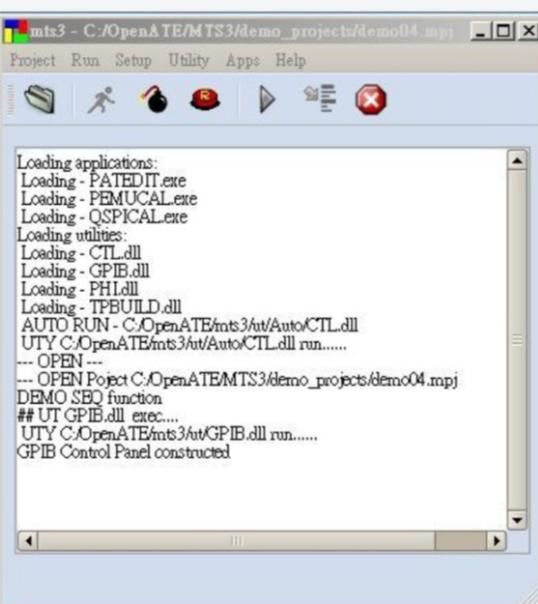
# Download and Install MTS3

- Download MTS3 install file from OpenATE official web
  - <http://www.openate.com/>



## MTS3 TEST PLATFORM

MTS3 is a compact software platform that includes test program development, test operation and debug functionalities. Based on open architectural design, this platform can run on any hardware on Windows and can easily add new instruments API and utilities.



The screenshot shows the MTS3 software interface with a title bar "mts3 - C:/OpenATE/MTS3/demo\_projects/demo04.mpj". The menu bar includes Project, Run, Setup, Utility, Apps, and Help. Below the menu is a toolbar with icons for file operations. The main window contains a log window titled "Loading applications:" which lists several files being loaded: PATEDIT.exe, PEMUCALExe, QSPICALExe, CTL.dll, GPIB.dll, PH.dll, and TPBUILD.dll. It also shows an "AUTO RUN" section with paths to C:/OpenATE/mts3/ut/Auto/CTL.dll and UTY C:/OpenATE/mts3/ut/Auto/CTL.dll run..... followed by "OPEN" and "OPEN Project" sections. The log ends with "GPIB Control Panel constructed".

**DOWNLOAD**

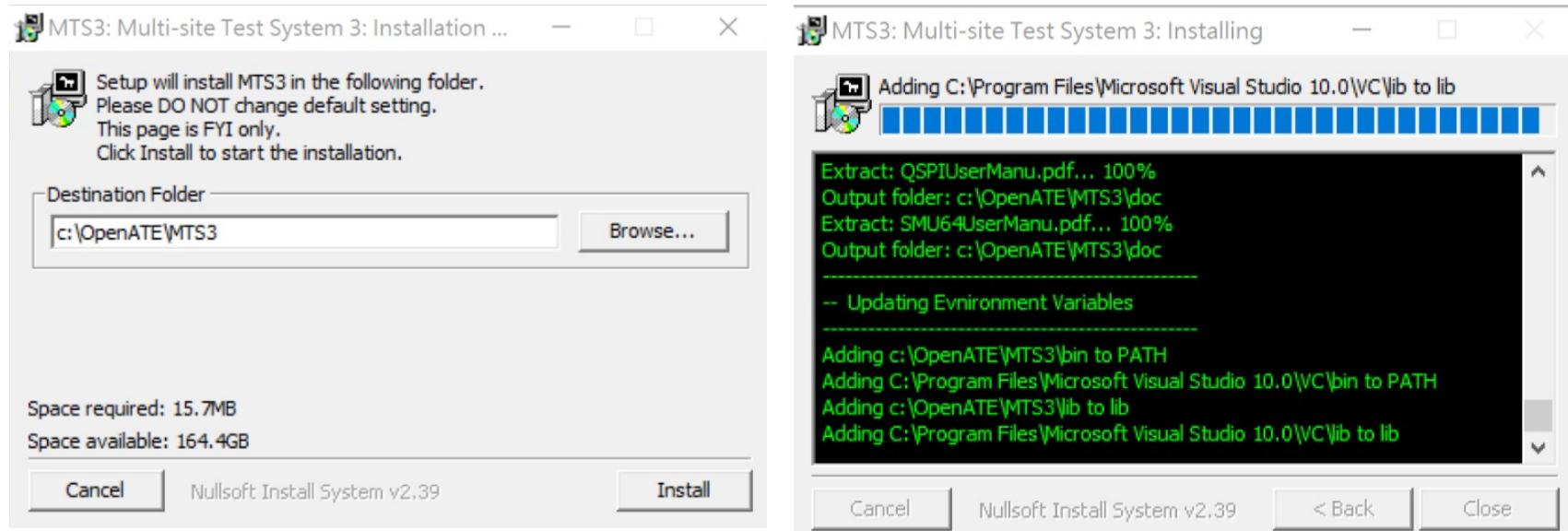
---

1 [Data sheet](#)  
2 [Install](#)

[CONTACT US](#)

1. Data sheet:  
User can download MTS3 data sheet, including simple using instructions.
2. Install:  
Download MTS3 install file from this link.

- Step 1:  
Click and install mts3\_VISAinstall201xxxxx.exe



- Step 2  
Install Compiler "**Visual C++ 2013 Express**"  
Complete the following instructions after installing the VC++:  
(a) Add "C:\Program Files (x86)\Adobe\Reader11.0\Reader" to PATH.

- Step 3: Download "**NI-VISA**" and install

# Introduction to MTS3

# Documents

- MTS3 user manual
  - C:\OpenATE\MTS3\doc\mts3\_user.manual.pdf
  - C:\OpenATE\MTS3\doc\MTS3\_install\_readme.pdf

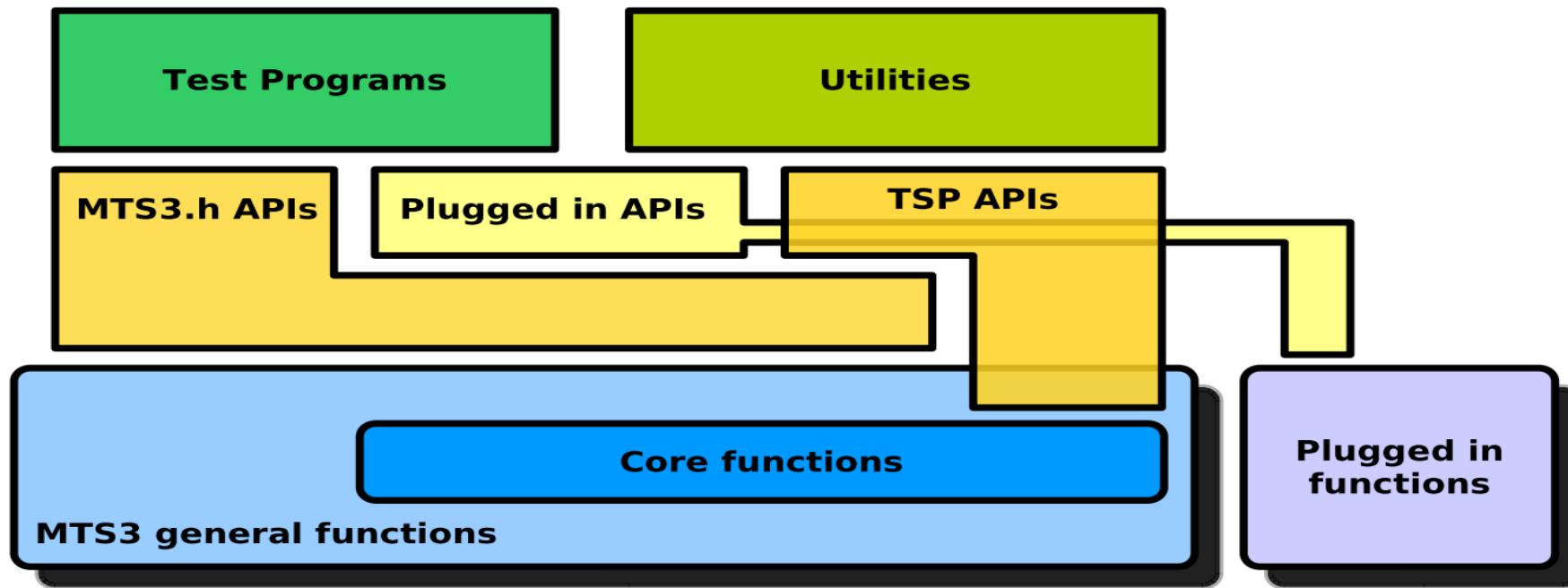
# Features

- ATE software platform which can run with any hardware on Windows.
- Reconfigurable for new instruments API
- C/C++ development environments
- Integrated debugging capabilities:
  - DATALOG , DEBUG, OVERRIDE, STOP ON\_FAIL
- Loop test capability
- Summary report ready

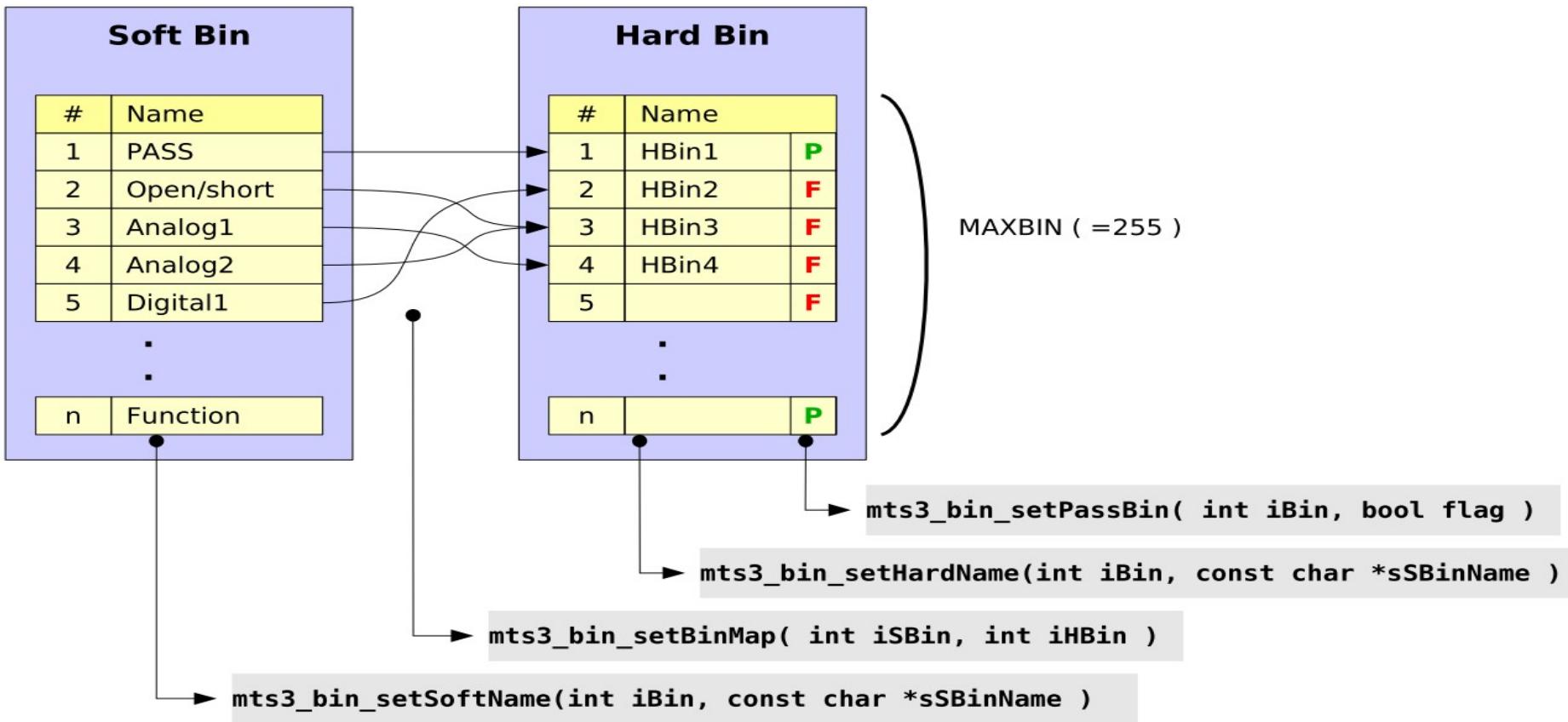
# Features (cnt.)

- Datalog to file or STDF data format
- Multi-sites testing capability
  - No need to modify test program
  - One single test program for multi-sites testing
- Reconfigurable prober / handler interface.
- Easy to connect any type of prober / handler form any peripheral supplier

# Software Architecture



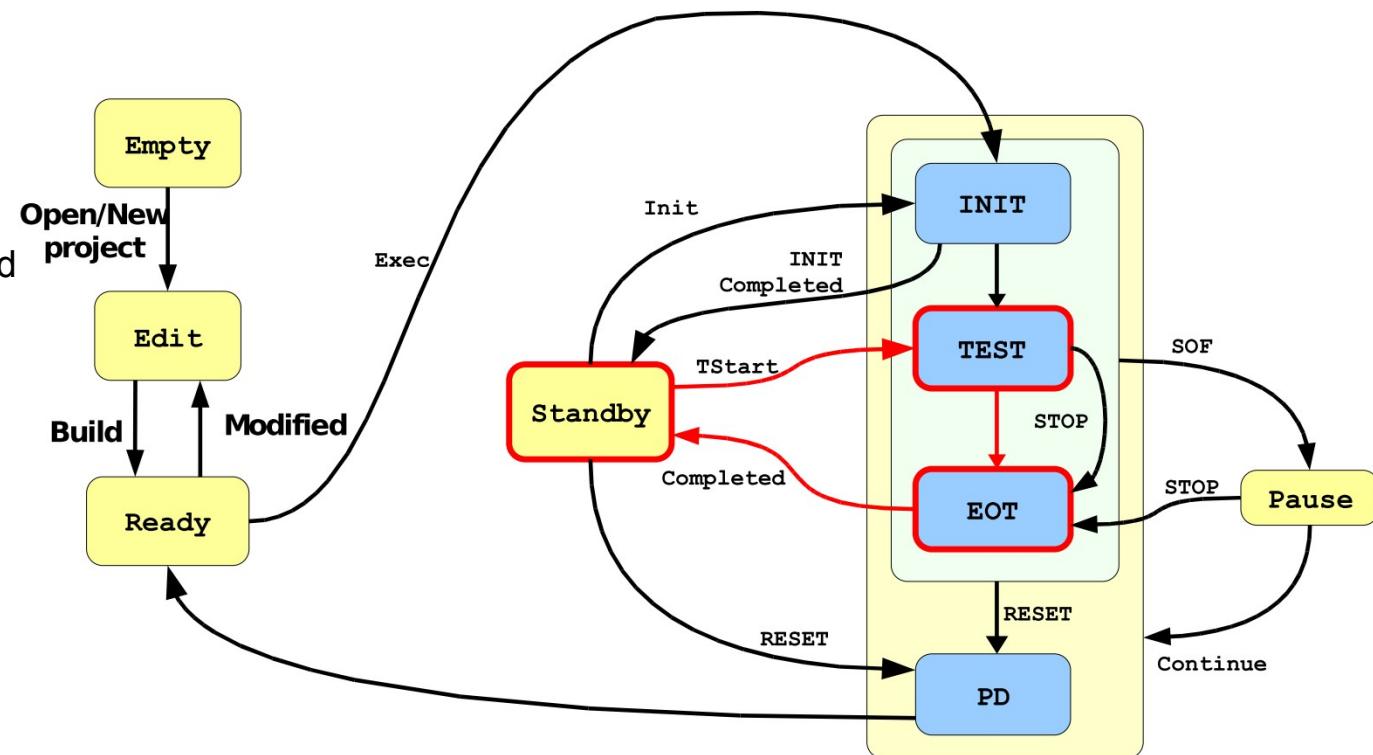
# Bin related functions



# State Diagram

The following is the state diagram for MTS3:

- When user opens or creates a project, Exec it, the INIT module will be invoked and then the system stays at Standby state and waiting for TStart from PHI.
- When MTS3 receives TStart, the TEST module will be invoked, then followed by EOT module, and then after TEST & EOT have completed execution, the system goes back to Standby again and waits for next Tstart.
- When RESET button is pressed by user during above operations, the PD will be invoked, and then the system goes to Ready state.



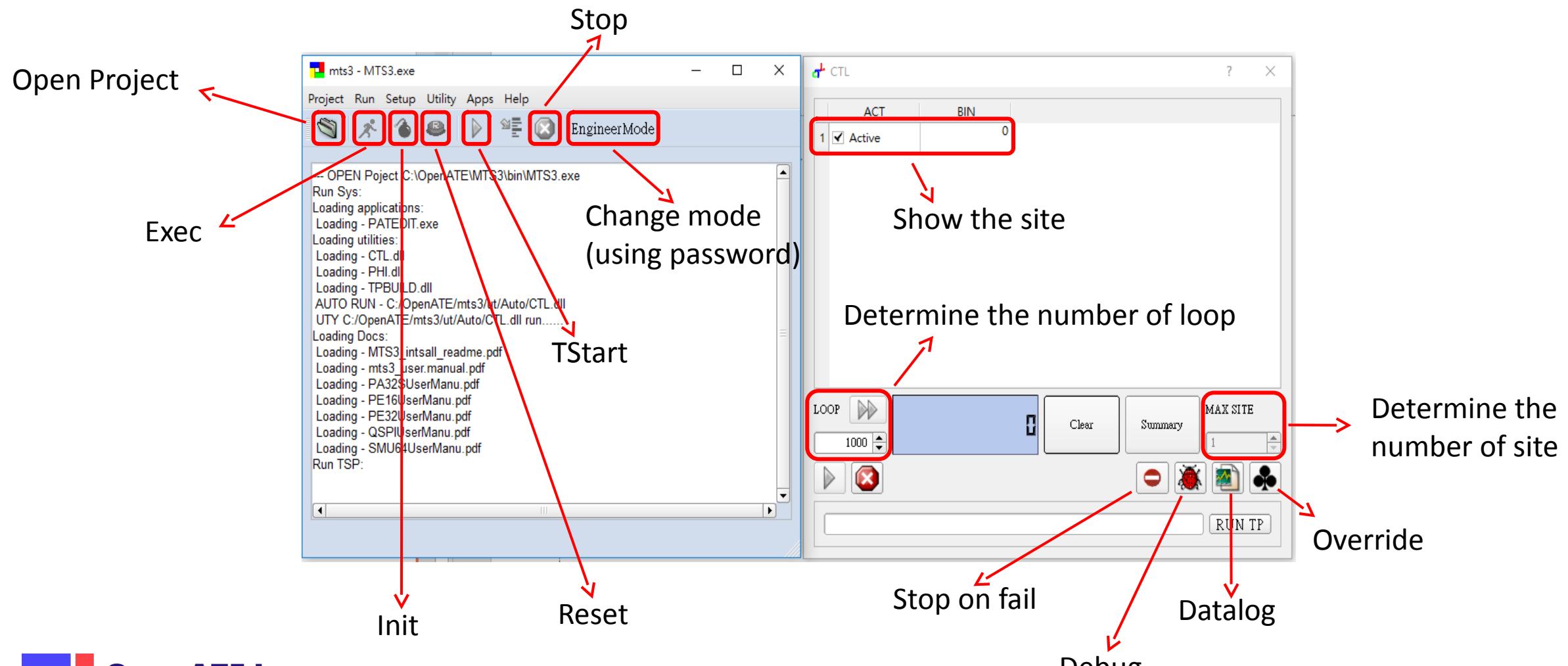
# Test program structure

- INIT
  - The INIT module is responsible for the initialization of the test program. User should place all the initial routines or start-up procedures in here.
- TEST
  - The TEST module is the main test sequence. User should put all the test statements that applied to DUT in this module. Device pass/fail decision and binning strategy should be placed in this module.

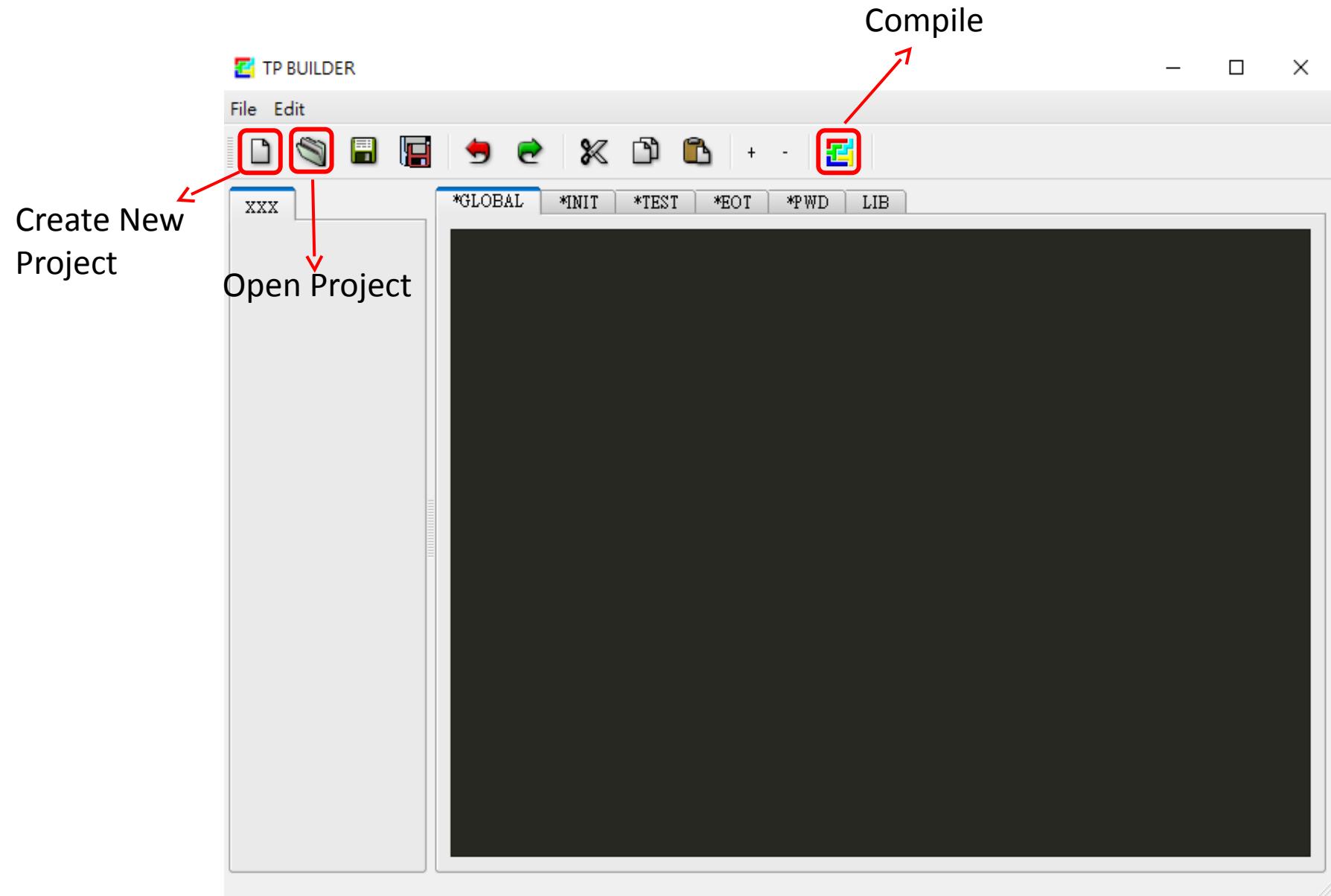
# Test program structure (cnt.)

- EOT
  - The EOT module will be called after TEST module is executed completed; so the EOT module also applied to each device under test.
- POWERDOWN
  - The POWERDOWN module (PD) is invoked when user want to leave the MTS3 system and hoping to turn off all instrument powers insure safety of the test system.

# User Interface for MTS3

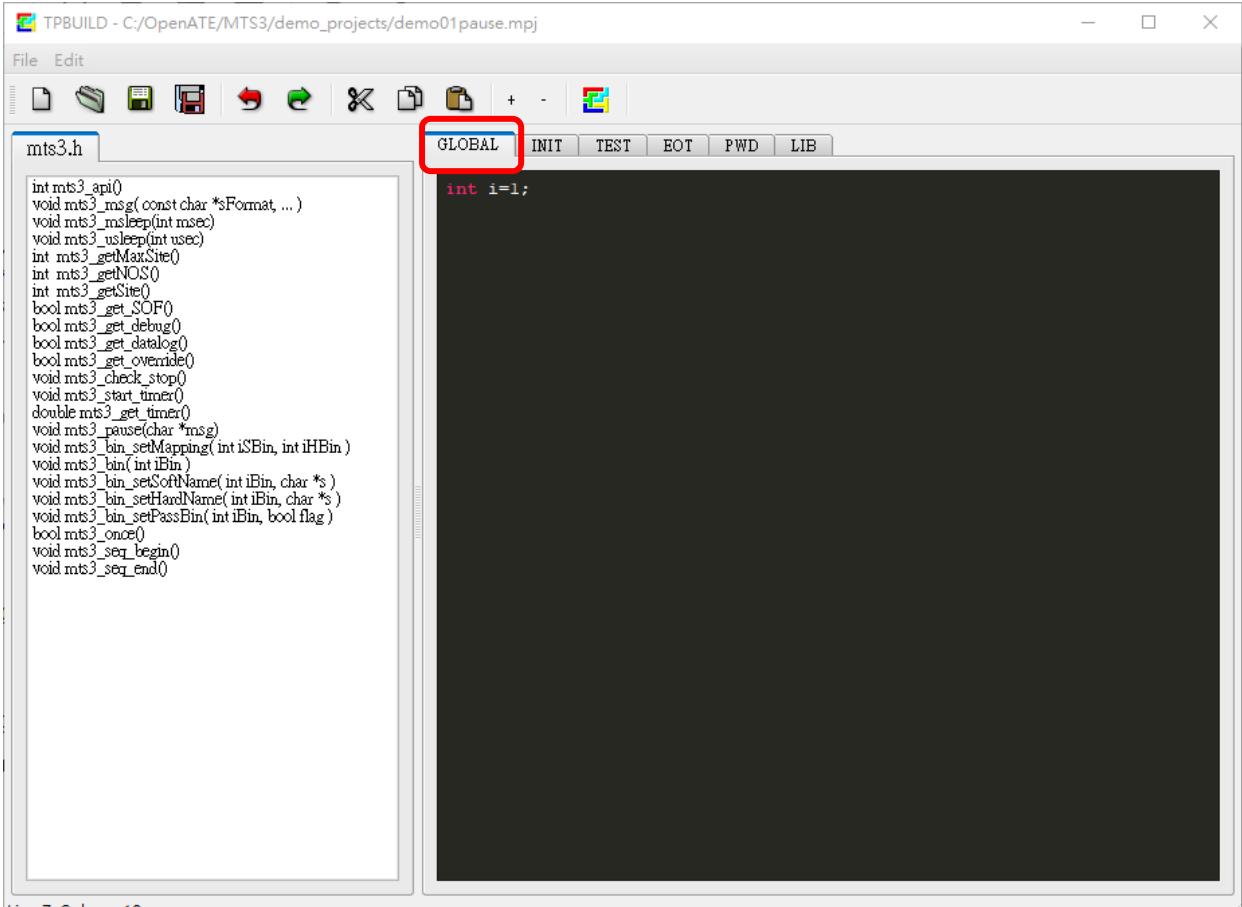


- Utility → TPBUILD



# GLOBAL

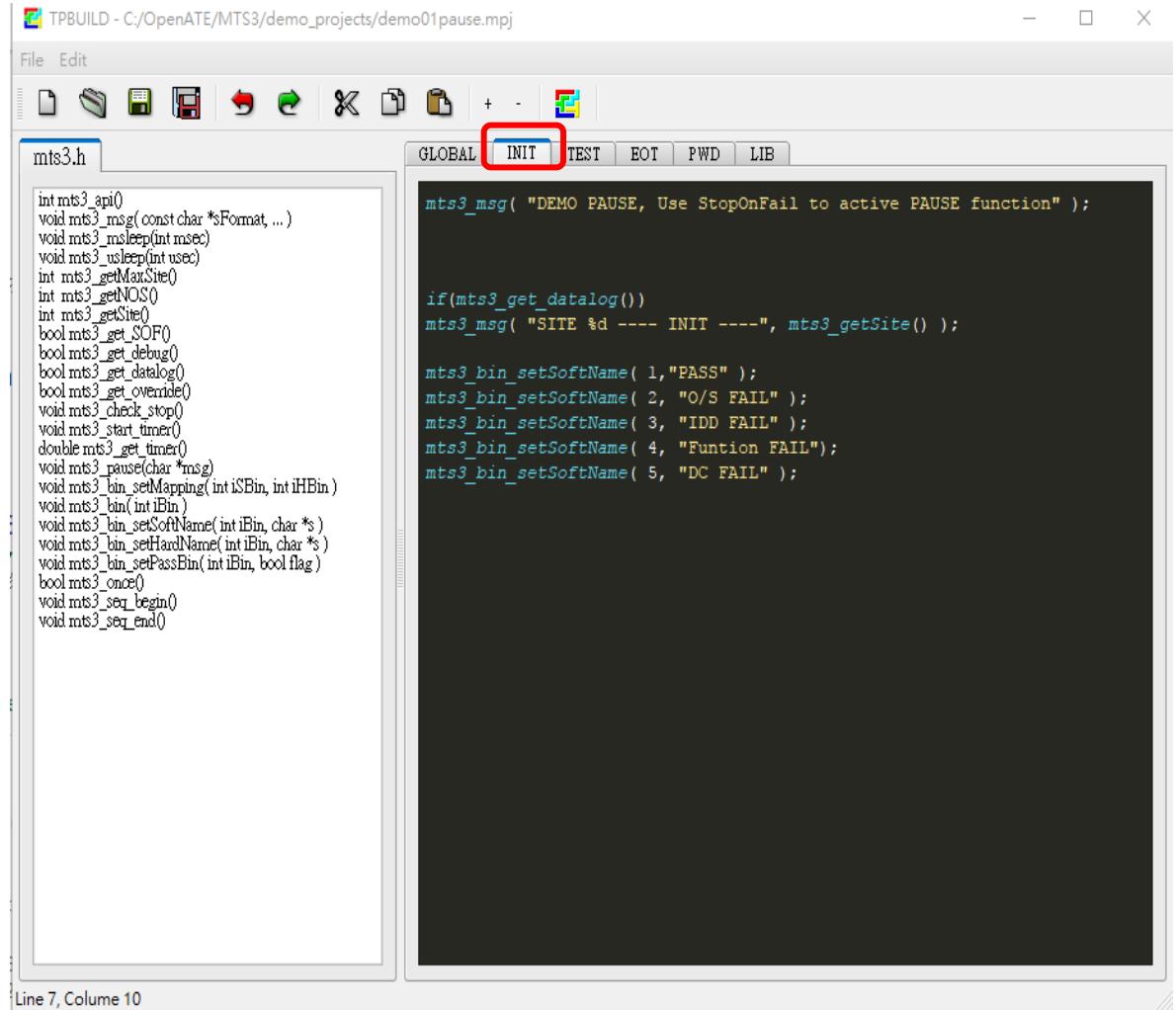
- Declare global variables and functions.



The screenshot shows a software interface titled "TPBUILD - C:/OpenATE/MTS3/demo\_projects/demo01pause.mpj". The window has a toolbar with various icons at the top. Below the toolbar, there are several tabs: "GLOBAL" (which is highlighted with a red box), "INIT", "TEST", "EOT", "PWD", and "LIB". The main area of the window is a code editor displaying the contents of a file named "mts3.h". The code in "mts3.h" includes declarations for various functions such as int mts3\_api(), void mts3\_msg(const char \*sFormat, ...), void mts3\_msleep(int msec), void mts3\_usleep(int usec), int mts3\_getMaxSite(), int mts3\_getNOS(), int mts3\_getSite(), bool mts3\_get\_SOF(), bool mts3\_get\_debug(), bool mts3\_get\_datalog(), bool mts3\_get\_overide(), void mts3\_check\_stop(), void mts3\_start\_timer(), double mts3\_get\_timer(), void mts3\_pause(char \*msg), void mts3\_bin\_setMapping(int iSBin, int iHBin), void mts3\_bin(int iBin), void mts3\_bin\_setSoftName(int iBin, char \*s), void mts3\_bin\_setHardName(int iBin, char \*s), void mts3\_bin\_setPassBin(int iBin, bool flag), bool mts3\_once(), void mts3\_seq\_begin(), and void mts3\_seq\_end(). A line of code "int i=1;" is also present. The status bar at the bottom of the code editor says "Line 7, Column 10".

# INIT

- We write initialization action in INIT window.
- When the “Exec” button or “Init” button is pressed, it will be initialized.



The screenshot shows the TPBUILD software interface with the title bar "TPBUILD - C:/OpenATE/MTS3/demo\_projects/demo01pause.mpj". The menu bar includes File, Edit, and several icons. Below the menu is a toolbar with various buttons. A tab bar at the top right has tabs for GLOBAL, INIT, TEST, EOT, PWD, and LIB, with the INIT tab highlighted by a red box. The main area contains two panes. The left pane displays the contents of "mts3.h" with a list of function prototypes. The right pane shows the source code for the INIT section, which includes a message to the user and calls to various initialization functions. The status bar at the bottom indicates "Line 7, Column 10".

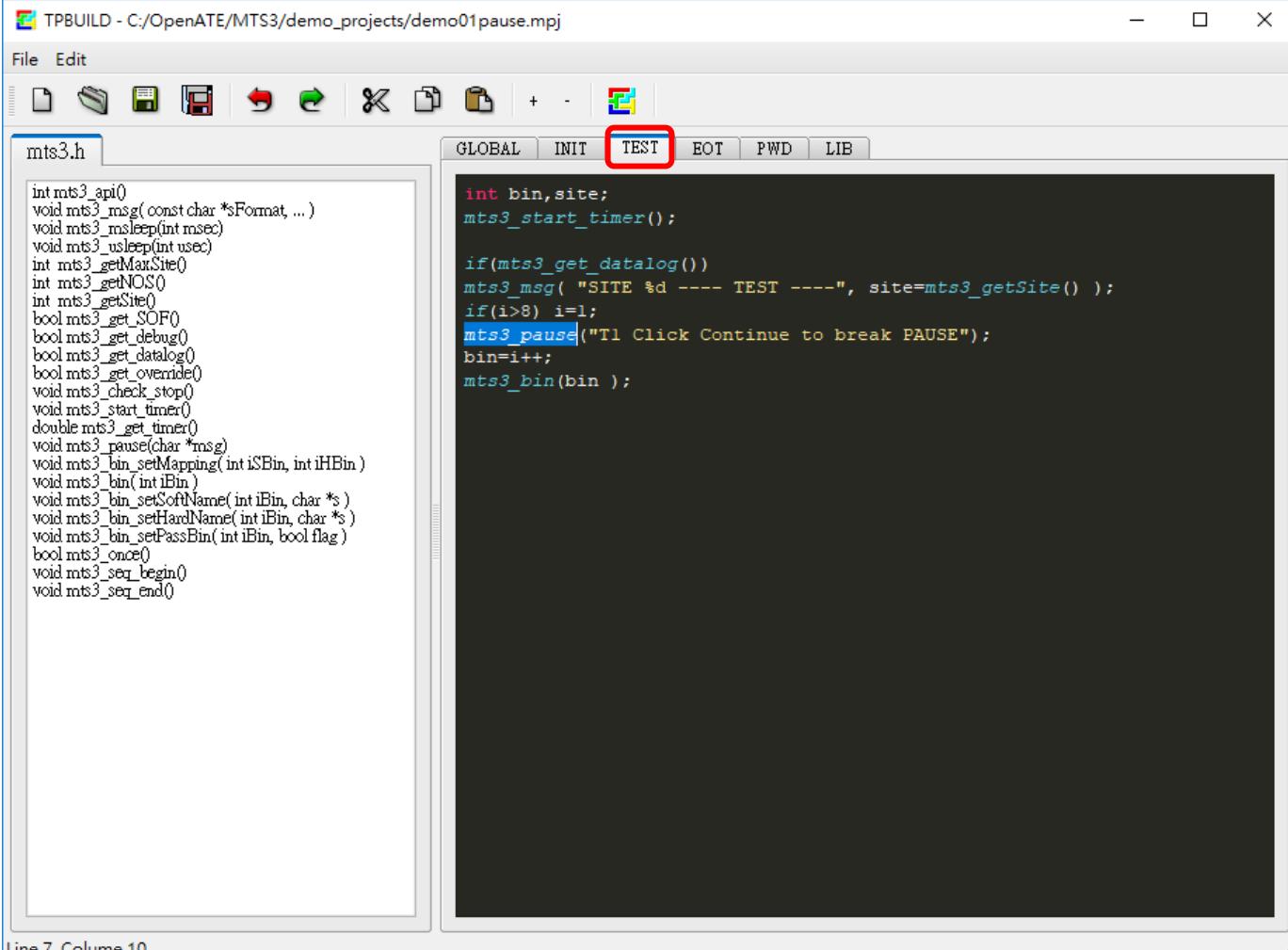
```
mts3.h
int mts3_api()
void mts3_msg( const char *sFormat, ... )
void mts3_msleep(int msec)
void mts3_usleep(int usec)
int mts3_getMaxSite()
int mts3_getNOS()
int mts3_getSite()
bool mts3_get_SOF0()
bool mts3_get_debug()
bool mts3_get_datalog()
bool mts3_get_override()
void mts3_check_stop()
void mts3_start_timer()
double mts3_get_timer()
void mts3_pause(char *msg)
void mts3_bin_setMapping( int iSBin, int iHBin )
void mts3_bin_set( int iBin )
void mts3_bin_setSoftName( int iBin, char *s )
void mts3_bin_setHardName( int iBin, char *s )
void mts3_bin_setPassBin( int iBin, bool flag )
bool mts3_once()
void mts3_seq_begin()
void mts3_seq_end()

mts3_msg( "DEMO PAUSE, Use StopOnFail to active PAUSE function" );
if(mts3_get_datalog())
mts3_msg( "SITE %d ---- INIT ----", mts3_getSite() );

mts3_bin_setSoftName( 1,"PASS" );
mts3_bin_setSoftName( 2, "O/S FAIL" );
mts3_bin_setSoftName( 3, "IDD FAIL" );
mts3_bin_setSoftName( 4, "Funtion FAIL" );
mts3_bin_setSoftName( 5, "DC FAIL" );
```

# TEST

- We write test programs in TEST window.
- When the “Tstart” button is pressed, it will execute the program in TEST window.



The screenshot shows the TPBUILD software interface with the following details:

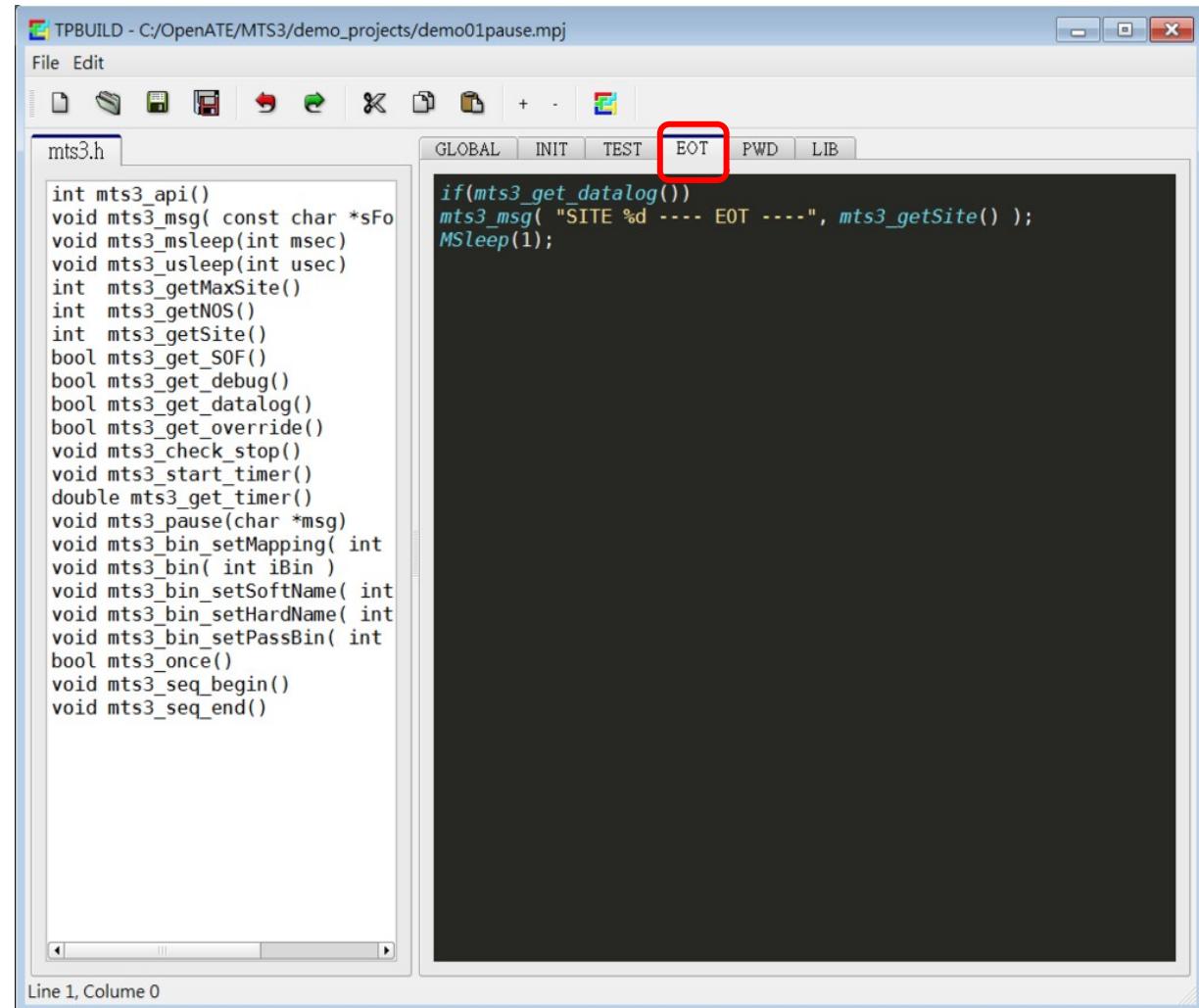
- Title Bar:** TPBUILD - C:/OpenATE/MTS3/demo\_projects/demo01pause.mpj
- Menu Bar:** File Edit
- Toolbar:** Standard toolbar icons.
- Tab Bar:** GLOBAL INIT TEST EOT PWD LIB. The TEST tab is highlighted with a red box.
- Left Panel:** A code editor window titled "mts3.h" containing a list of C-style API functions. The functions listed include: int mts3\_api(), void mts3\_msg(const char \*sFormat, ...), void mts3\_msleep(int msec), void mts3\_usleep(int usec), int mts3\_getMaxSite(), int mts3\_getNOS(), int mts3\_getSite(), bool mts3\_get\_SOF0(), bool mts3\_get\_debug(), bool mts3\_get\_datalog(), bool mts3\_get\_override(), void mts3\_check\_stop(), void mts3\_start\_timer(), double mts3\_get\_timer(), void mts3\_pause(char \*msg), void mts3\_bin\_setMapping(int iSBin, int iHBin), void mts3\_bin\_(int iBin), void mts3\_bin\_setSoftName(int iBin, char \*s), void mts3\_bin\_setHardName(int iBin, char \*s), void mts3\_bin\_setPassBin(int iBin, bool flag), bool mts3\_once(), void mts3\_seq\_begin(), void mts3\_seq\_end().
- Right Panel:** A code editor window titled "demo01pause.mpj" containing a C program. The code includes:

```
int bin,site;
mts3_start_timer();

if(mts3_get_datalog())
    mts3_msg( "SITE %d ---- TEST ----", site=mts3_getSite() );
if(i>8) i=1;
mtss3_pause("T1 Click Continue to break PAUSE");
bin=i++;
mtss3_bin(bin );
```
- Status Bar:** Line 7, Column 10

# EOT

- The EOT module will be called after TEST module is executed completed.
- So the EOT module also applied to each device under test.



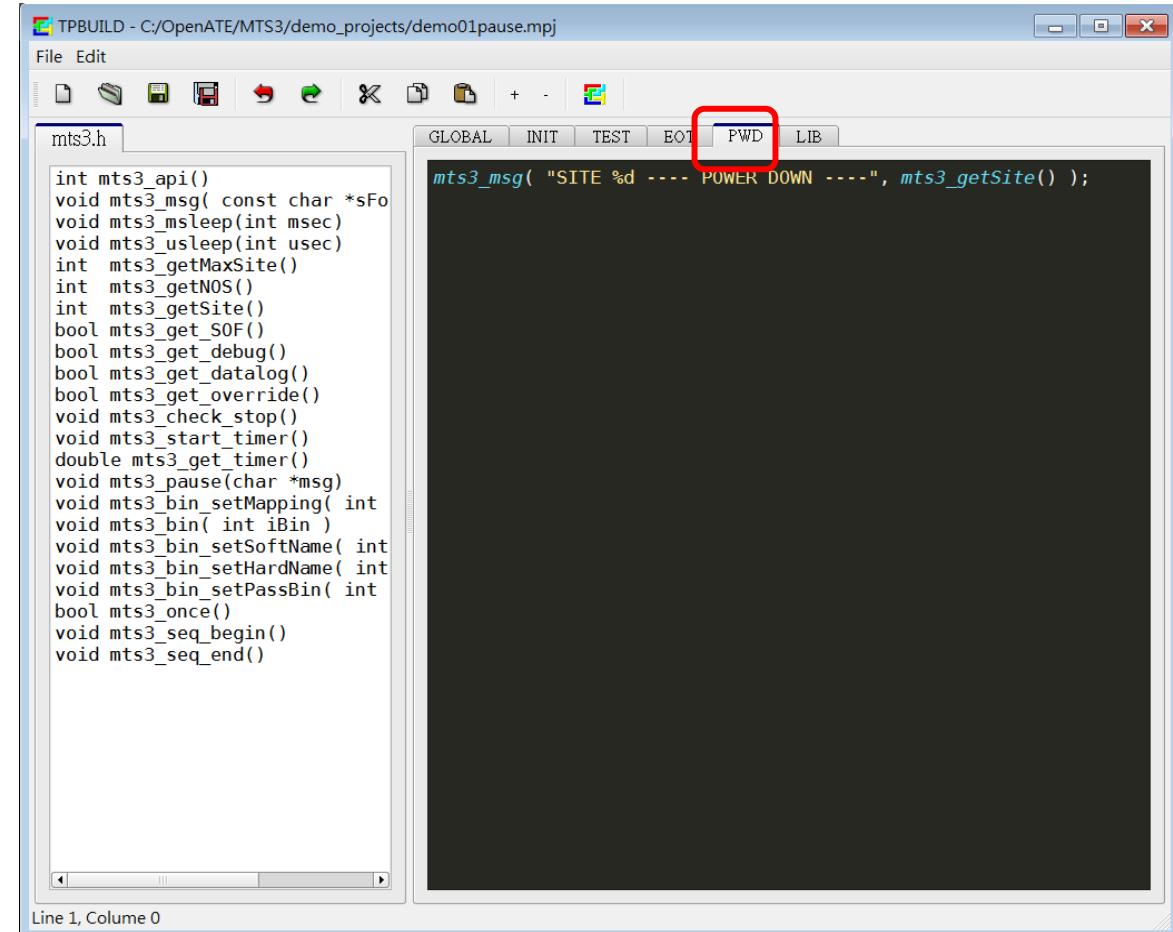
The screenshot shows a software interface titled "TPBUILD - C:/OpenATE/MTS3/demo\_projects/demo01pause.mpj". The window has tabs at the top: GLOBAL, INIT, TEST, EOT (which is highlighted with a red box), PWD, and LIB. On the left, there is a code editor window titled "mts3.h" containing a list of function prototypes. On the right, there is another code editor window showing a snippet of C code. The code in the right window includes an if-statement that checks if the mts3\_get\_datalog() function returns true, and if so, it prints a message to the console: "SITE %d ---- EOT ----", followed by the result of mts3\_getSite() and a one-second sleep using MSleep(1);

```
int mts3_api()
void mts3_msg( const char *sFo
void mts3_msleep(int msec)
void mts3_usleep(int usec)
int mts3_getMaxSite()
int mts3_getNOS()
int mts3_getSite()
bool mts3_get_SOF()
bool mts3_get_debug()
bool mts3_get_datalog()
bool mts3_get_override()
void mts3_check_stop()
void mts3_start_timer()
double mts3_get_timer()
void mts3_pause(char *msg)
void mts3_bin_setMapping( int
void mts3_bin( int iBin )
void mts3_bin_setSoftName( int
void mts3_bin_setHardName( int
void mts3_bin_setPassBin( int
bool mts3_once()
void mts3_seq_begin()
void mts3_seq_end()

if(mts3_get_datalog())
    mts3_msg( "SITE %d ---- EOT ----", mts3_getSite() );
    MSleep(1);
```

# PWD

- The POWERDOWN module (PD) is invoked when user wants to quit the MTS3 test program(RESET pressed) and turn off all instrument powers to insure safety of the test system.



The screenshot shows the TPBUILD software interface with the following details:

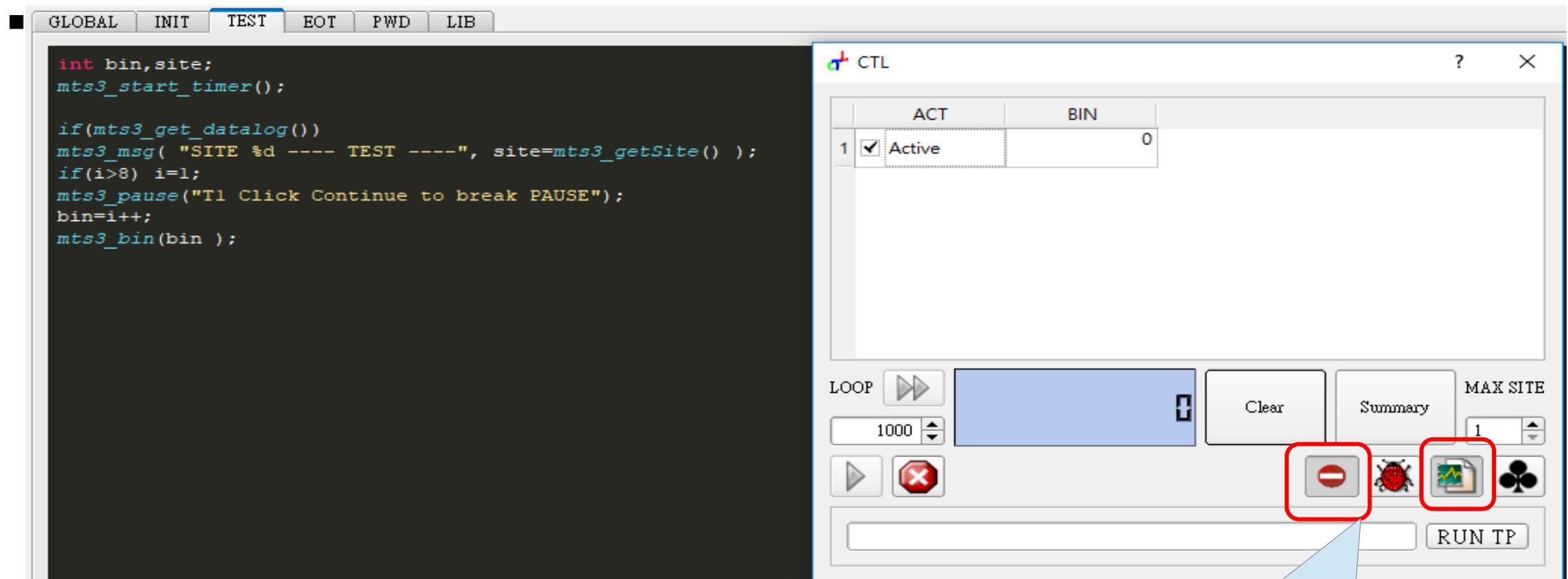
- Title Bar:** TPBUILD - C:/OpenATE/MTS3/demo\_projects/demo01pause.mpj
- Menu Bar:** File Edit
- Toolbars:** Standard toolbar (File, Open, Save, etc.)
- Tab Bar:** GLOBAL INIT TEST EOT **PWD** LIB (The PWD tab is highlighted with a red box)
- Left Panel:** A code editor window titled "mts3.h" containing the MTS3 API declarations. The declarations include functions like mts3\_api(), mts3\_msg(), mts3\_msleep(), mts3\_usleep(), mts3\_getMaxSite(), mts3\_getNOS(), mts3\_getSite(), mts3\_get\_SOF(), mts3\_get\_debug(), mts3\_get\_datalog(), mts3\_get\_override(), mts3\_check\_stop(), mts3\_start\_timer(), mts3\_get\_timer(), mts3\_pause(), mts3\_bin\_setMapping(), mts3\_bin(), mts3\_bin\_setSoftName(), mts3\_bin\_setHardName(), mts3\_bin\_setPassBin(), mts3\_once(), mts3\_seq\_begin(), and mts3\_seq\_end().
- Right Panel:** A large black panel representing the code editor's workspace.
- Status Bar:** Line 1, Column 0

# Demo Project1

Step-by-step

# Demo Project\_1

## Demo PAUSE



# Introduction to API

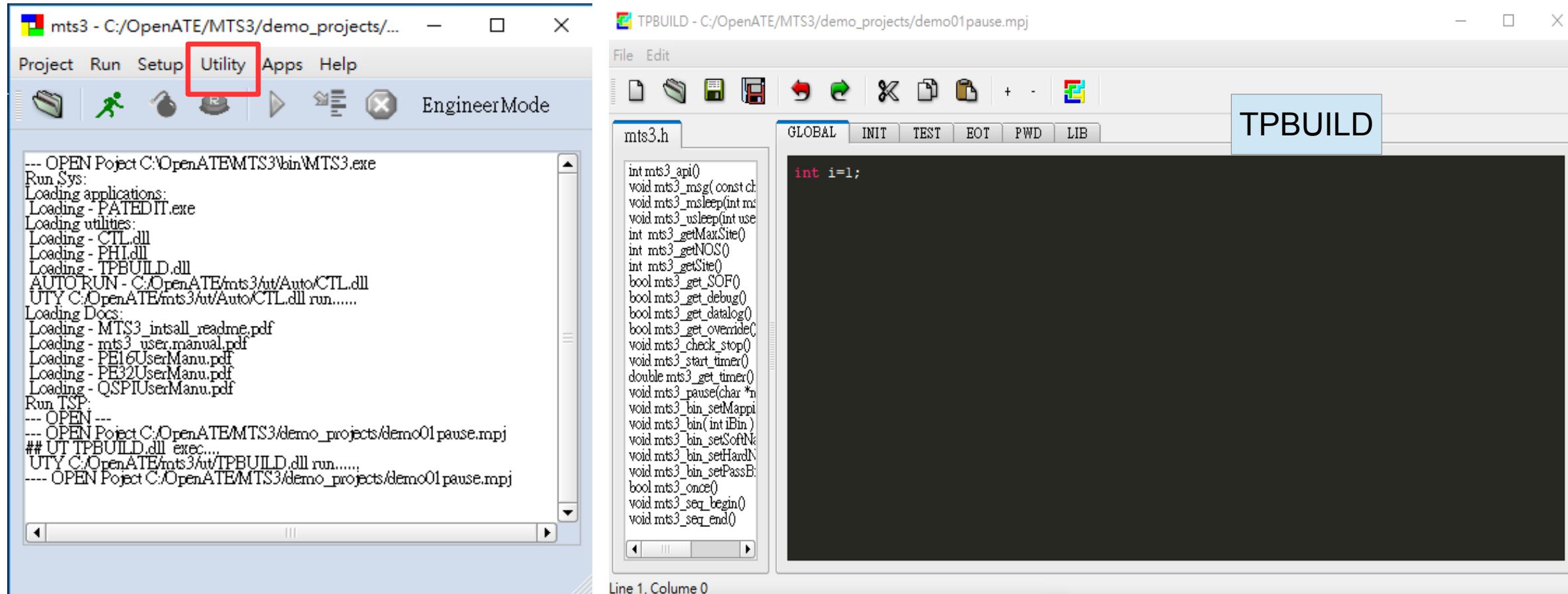
- `void mts3_msg( const char *sFormat, ... );`
  - Print message.
  - Usage: arguments are same as c standard function “printf”.
- `bool mts3_get_datalog();`
  - Return button 'Datalog' state of main dialog.
- `void mts3_bin_setSoftName( int iBin, char *s );`
  - Set name of specified software bin.

# Introduction to API

- `void mts3_pause(char *msg);`
  - Pause execution of test program until user hits continue button.
- `void mts3_bin( int iBin );`
  - Issue bin for current test.

# Step1

- Open MTS3
- Open project “C:\OpenATE\MTS3\demo\_projects\demo01pause.mpj”
- Open TPBUILD <Utility → TPBUILD >



# Step2

- You can press “Datalog” button to see some information.

The image shows a software interface with two main windows. On the left is a code editor window with tabs: GLOBAL, INIT (which is selected), TEST, EOT, PWD, and LIB. The code in the INIT tab is:

```
GLOBAL INIT TEST EOT PWD LIB

mts3_msg( "DEMO PAUSE, Use StopOnFail to active PAUSE function" );

if(mts3_get_datalog())
    mts3_msg( "SITE %d ---- INIT ----", mts3_getSite() );

mts3_bin_setSoftName( 1,"PASS" );
mts3_bin_setSoftName( 2, "O/S FAIL" );
mts3_bin_setSoftName( 3, "IDD FAIL" );
mts3_bin_setSoftName( 4, "Funtion FAIL");
mts3_bin_setSoftName( 5, "DC FAIL" );
```

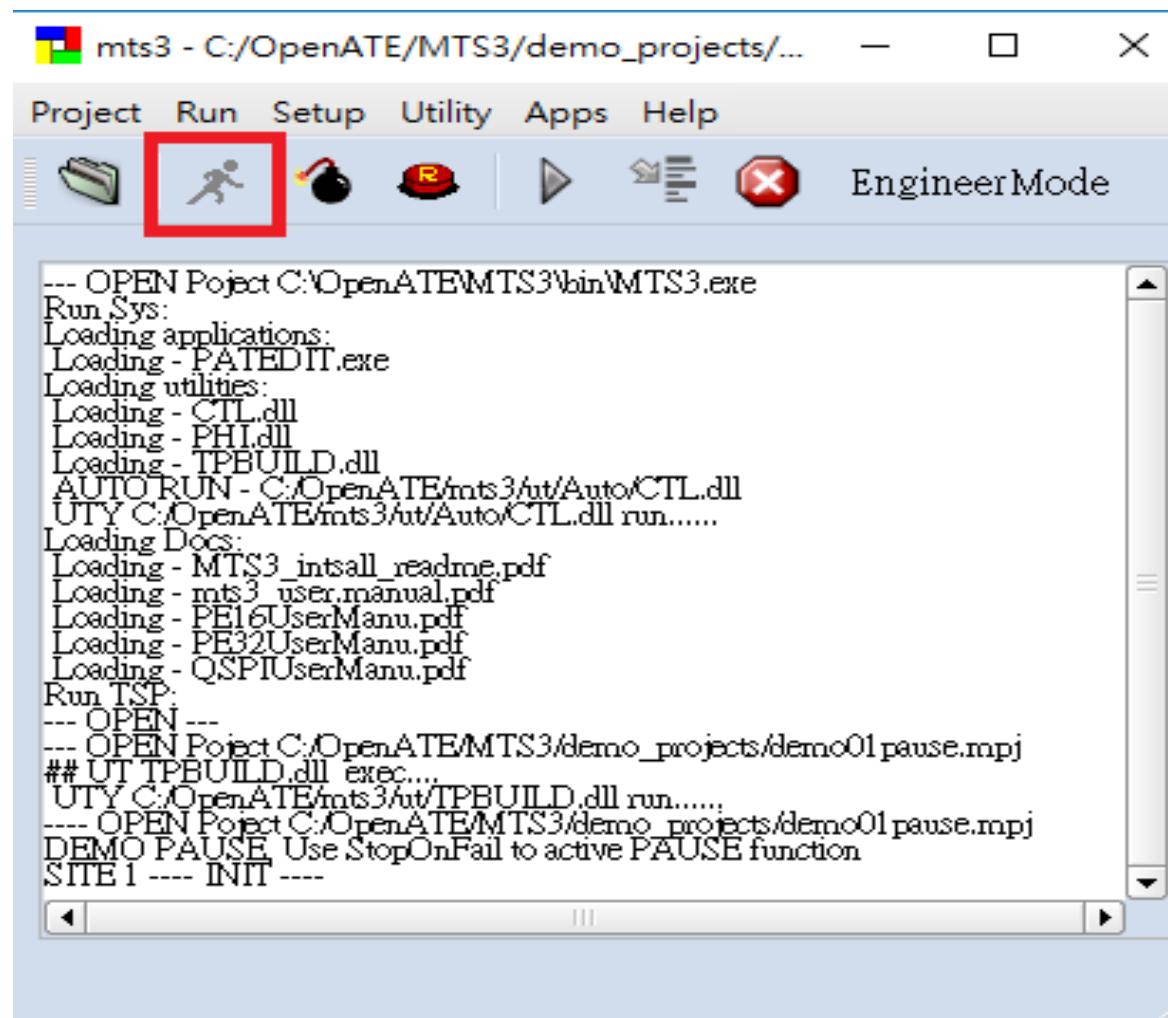
A red box highlights the first two lines of code: `if(mts3_get_datalog())` and `mts3_msg( "SITE %d ---- INIT ----", mts3_getSite() );`. A large blue arrow points from this highlighted code to the right window.

The right window is titled "CTL" and contains a table with two columns: ACT and BIN. The table has one row with ID 1, which has a checked checkbox in the ACT column and the value 0 in the BIN column. Below the table are buttons for LOOP, MAX SITE, Summary, and RUN TP. The RUN TP button is highlighted with a red box.

**When you press “Datalog” button,  
you can see information.**

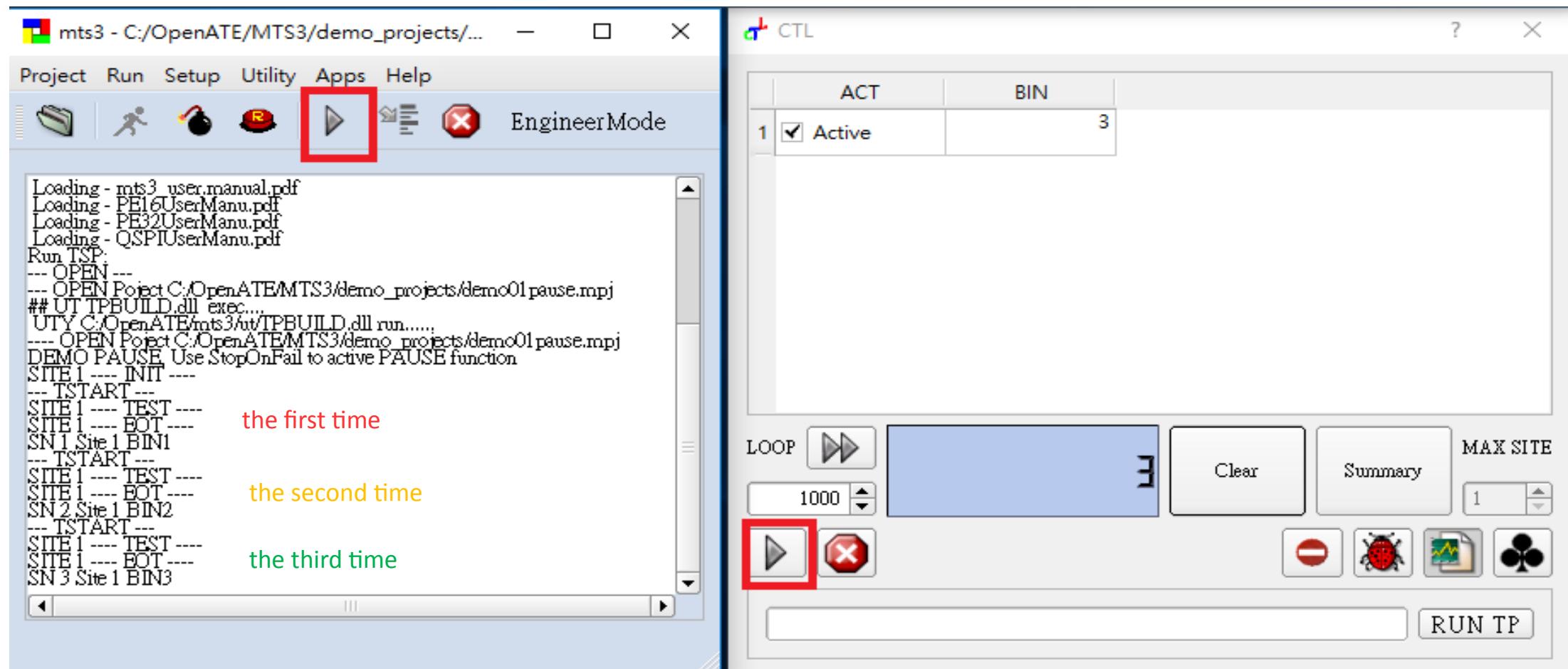
# Step3

- Press “Exec” button to initialize.



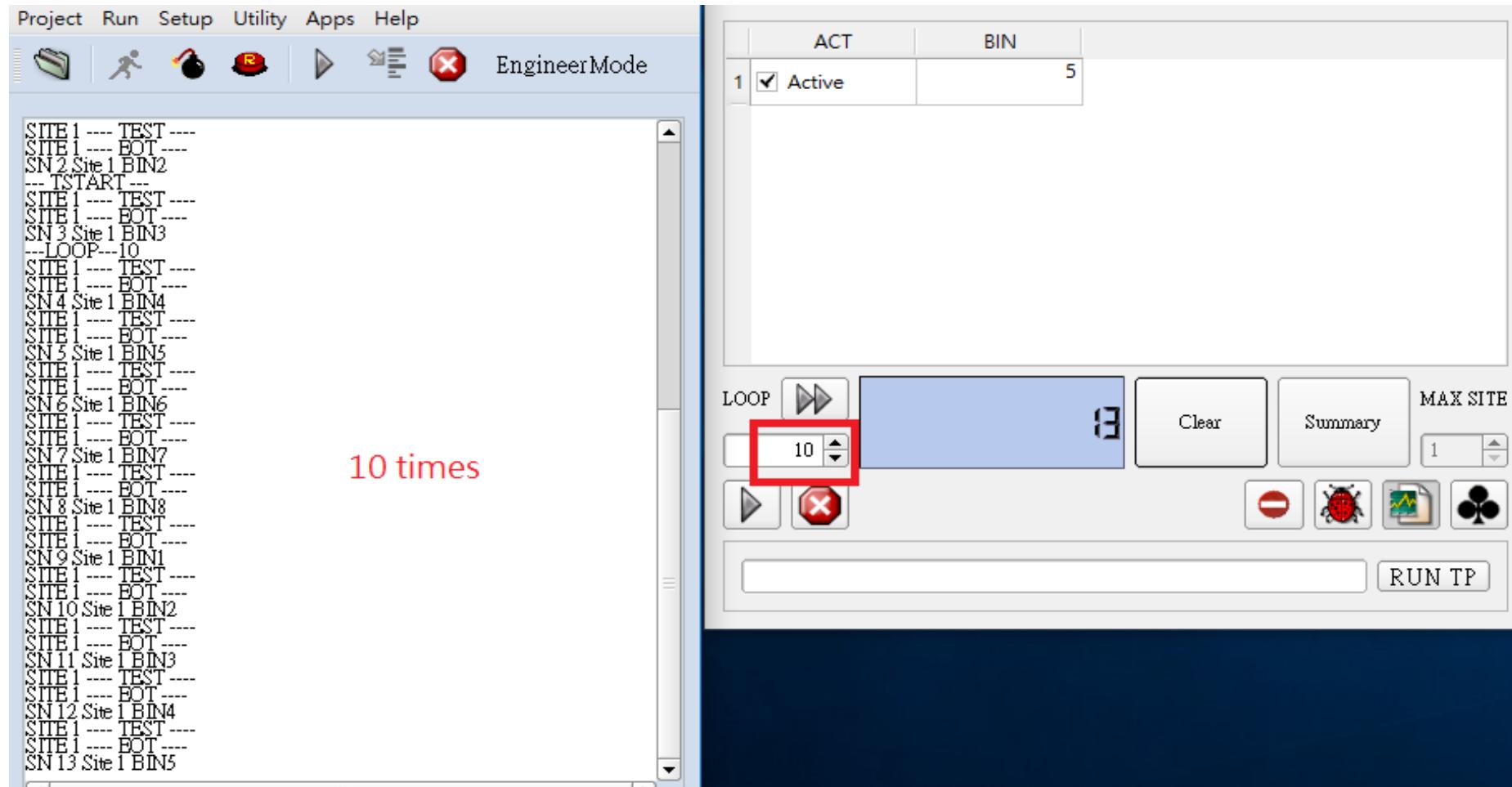
# Step4

- Press “Tstart” button to run the test program once.



# Step5

- You can press “Loop” button to run several times.



# Step6

- Now, we will test mts3\_pause() function.
- You must press “Pause” button and press “Tstart” button to run the test program once.

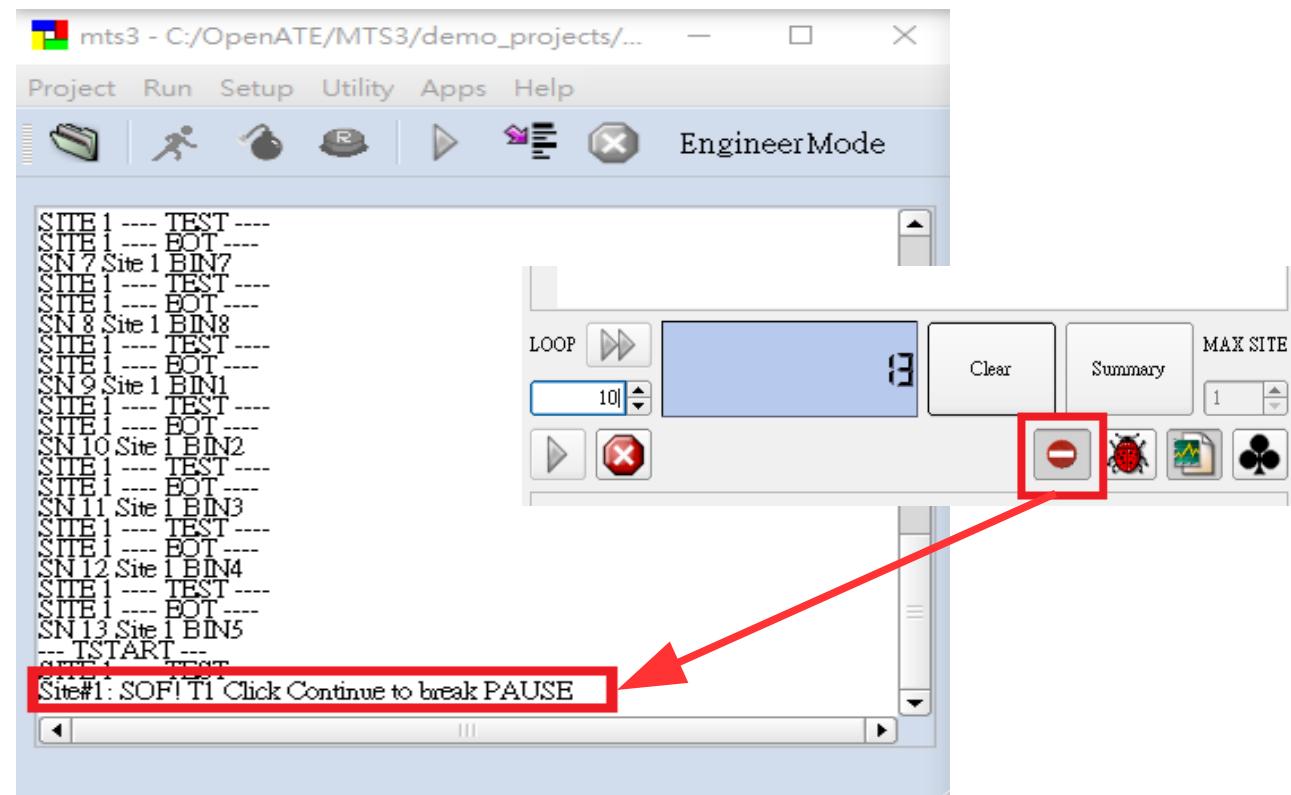
GLOBAL INIT TEST EOT PWD LIB

```
int bin,site;
mts3_start_timer();

if(mts3_get_datalog())
mts3_msg( "SITE %d ---- TEST ----", site=mts3_getSite() );
if(i>8) i=1;
mts3_pause("T1 Click Continue to break PAUSE");
bin++;
mts3_bin(bin );

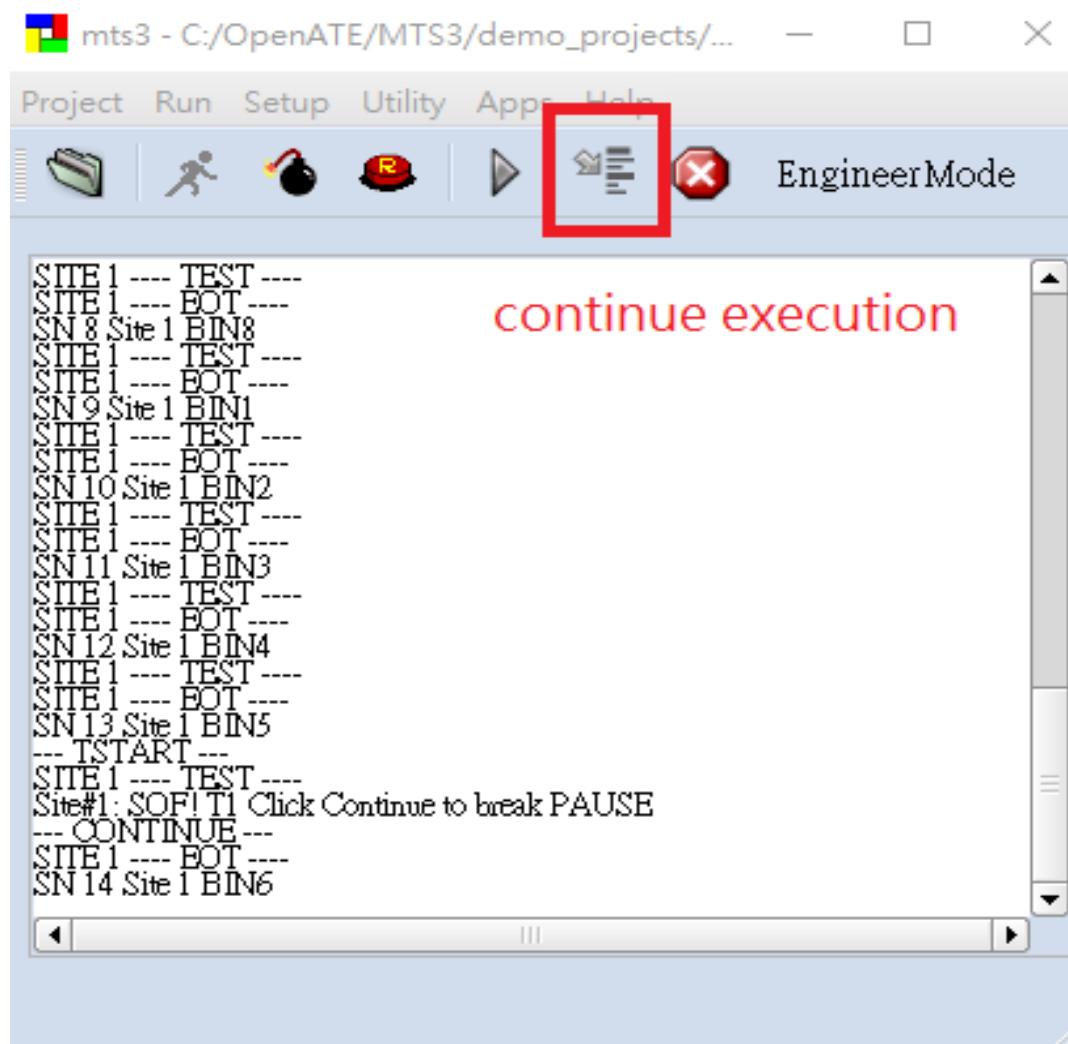

```

Pause execution of test program until user hits continue button.



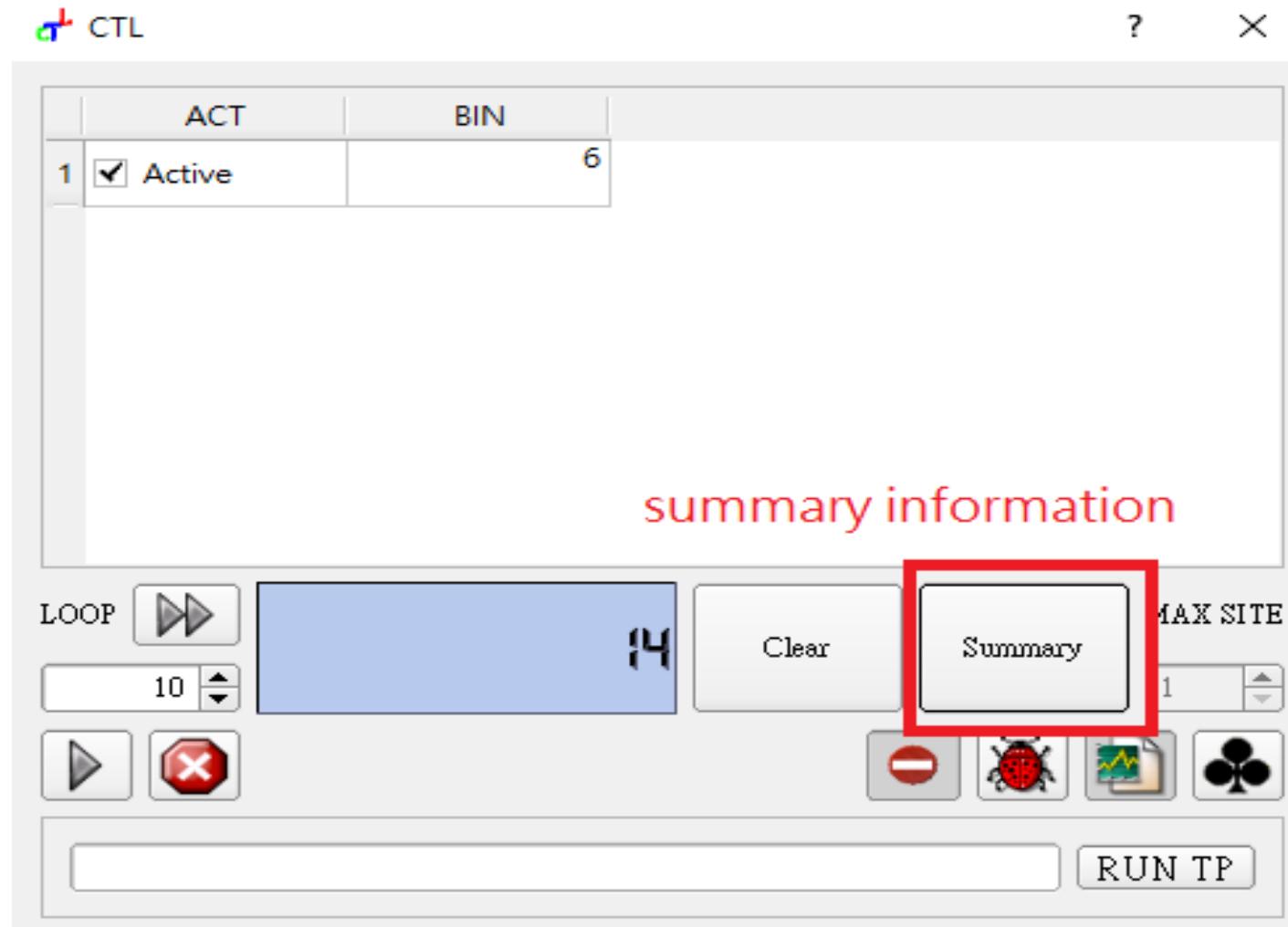
# Step7

- Press “Continue” button to continue execution.



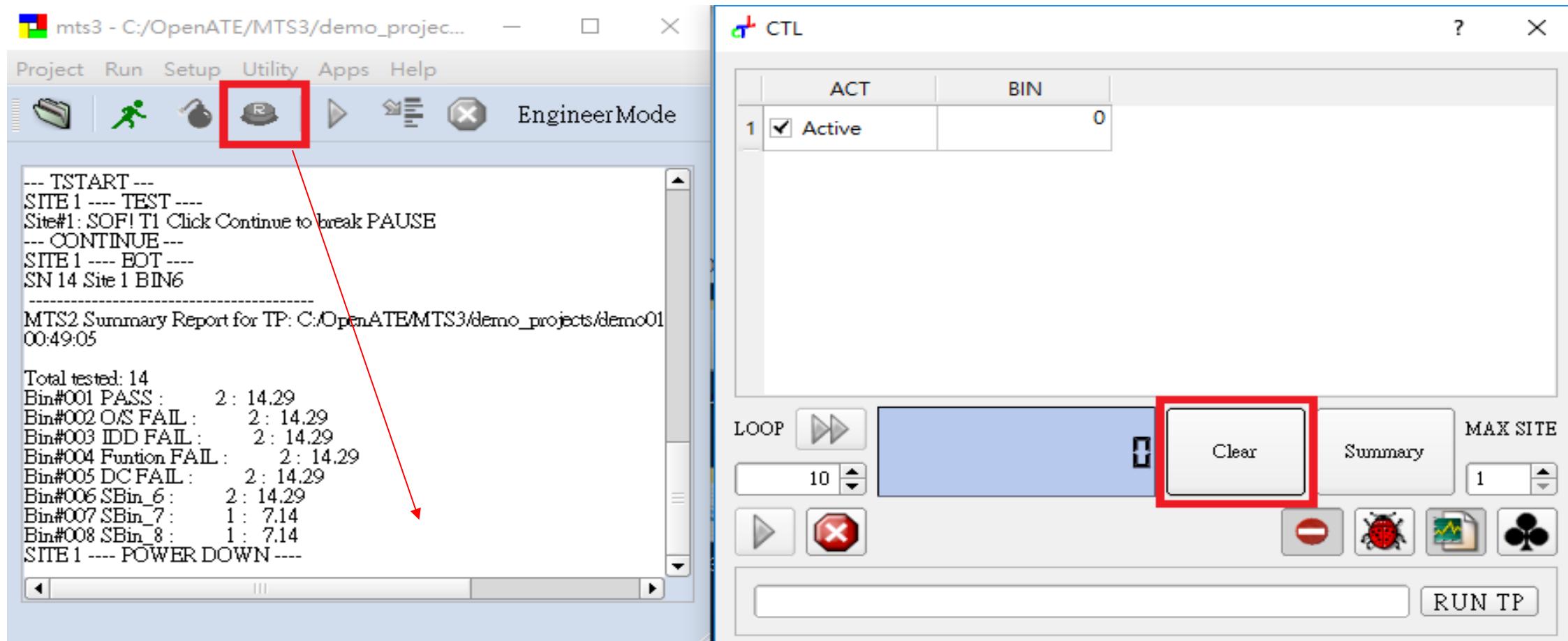
# Step8

- Press “Summary” button to see summary information.



# Step9

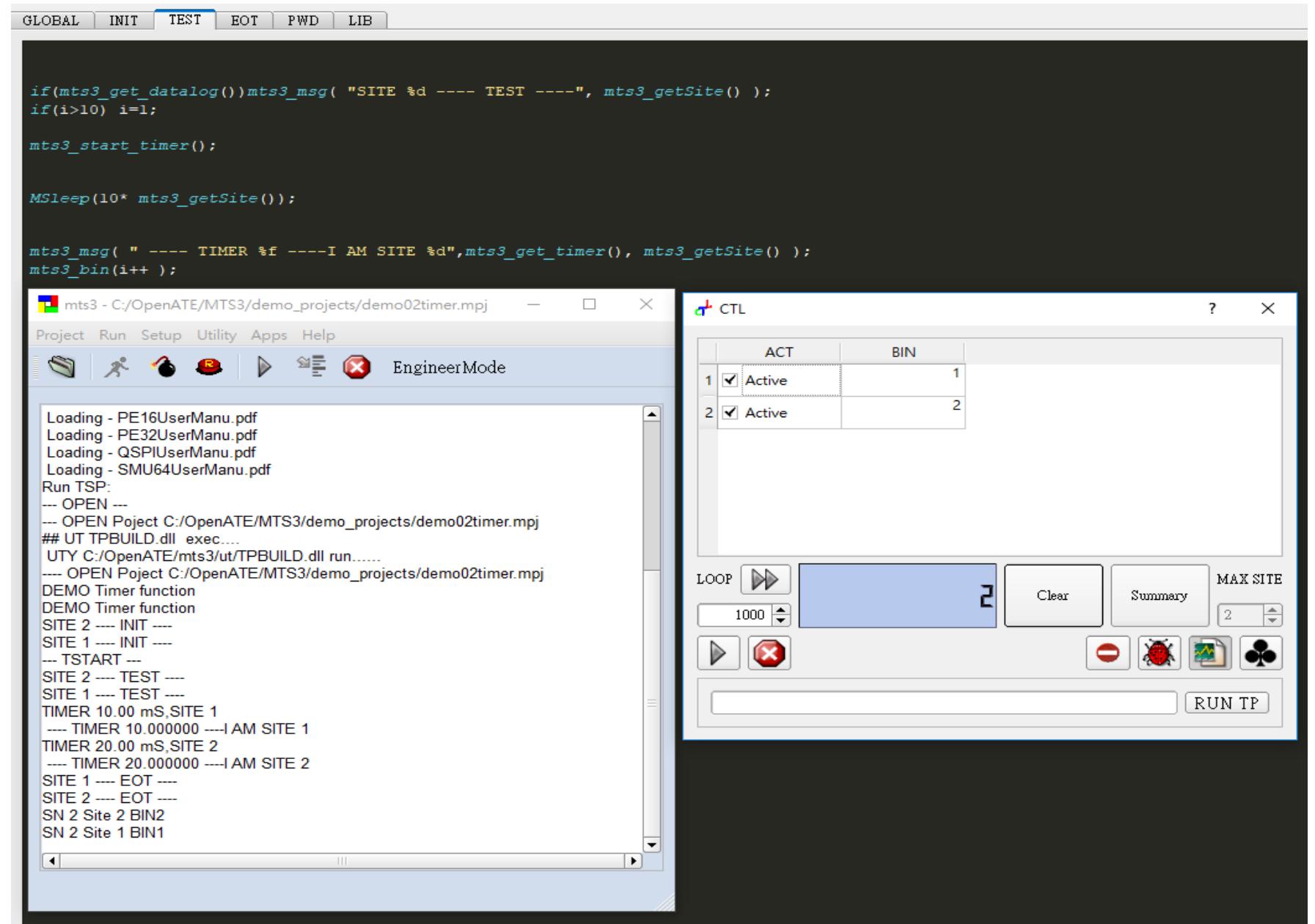
- Press “Reset” button to finish execution.
- Press “Clear” button to clear the number of times.



# Demo Project2

Step-by-step

# Demo TIMER



# Introduction to API

- `void mts3_start_timer();`
- `double mts3_get_timer();`
  - These two functions are used to measure the time elapsed from one point to another in the test program.
  - `mts3_start_time()` : Mark current time.
  - `mts3_get_timer()` : Return the time elapsed (in ms) since last call to `mts3_start_time()`.
  - If datalog is on, the time will also be printed in the datalog pane.

# Step1

- Open MTS3
- Open project “C:\OpenATE\MTS3\demo\_projects\demo02timer.mpj”
- Open TPBUILD <Utility → TPBUILD >

# Step2

- You can press “Datalog” button to see some information.

# Step3

- Press “Exec” button to initialize.

# Step4

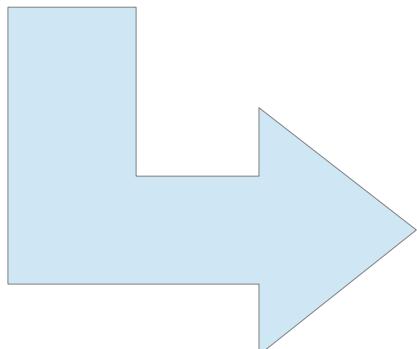
- Press “Tstart” button to run the test program once.

```
if(mts3_get_datalog()) mts3_msg( "SITE %d ---- TEST ----", mts3_getSite() );
if(i>10) i=1;

mts3_start_timer();

MSleep(10* mts3_getSite());

mts3_msg( " ---- TIMER %f ---- I AM SITE %d", mts3_get_timer(), mts3_getSite() );
mts3_bin(i++ );
```



mts3 - C:/OpenATE/MTS3/demo\_projects/demo02timer.mpj

Project Run Setup Utility Apps Help

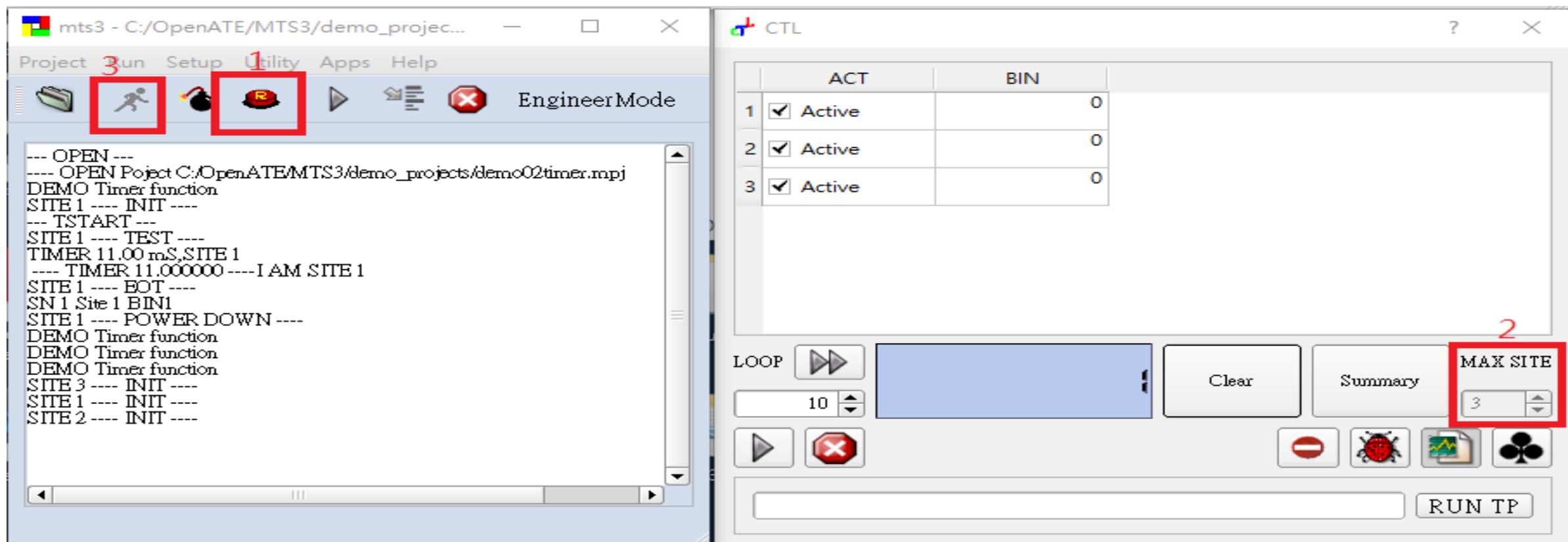
EngineerMode

--- OPEN ---  
--- OPEN Project C:/OpenATE/MTS3/demo\_projects/demo02timer.mpj  
DEMO Timer function  
SITE 1 ---- INIT ----  
--- TSTART ---  
SITE 1 ---- TEST ----  
TIMER 11.00 mS,SITE 1  
---- TIMER 11.000000 ----I AM SITE 1  
SITE 1 ---- EOT ----  
SN 1 Site 1 BIN1

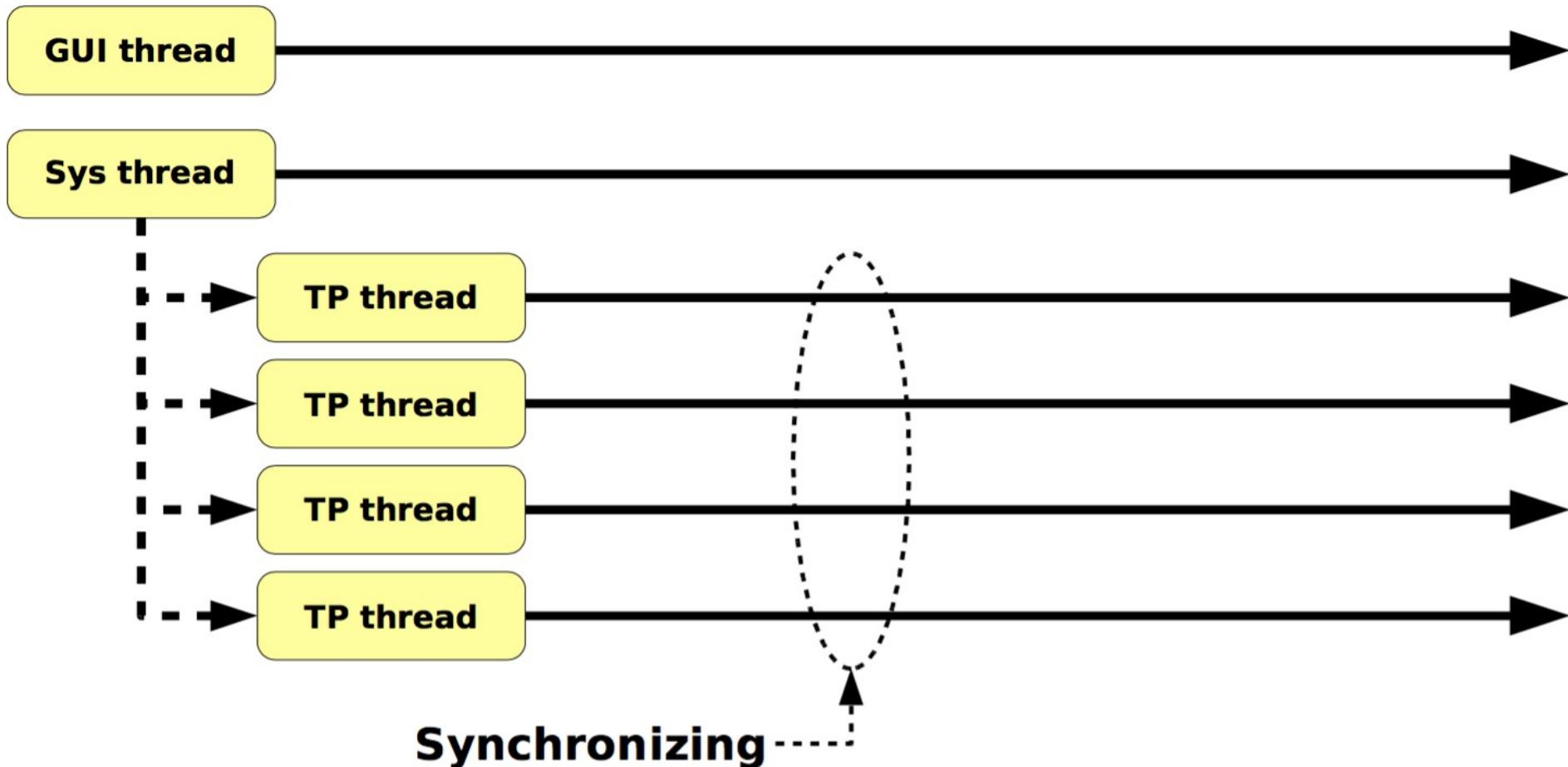
Site1 waits for 10 \* 1 ms

# Step5

- Press “Reset” button to finish execution.
- Increase the number of sites.
- Press “Exec” button to initialize.

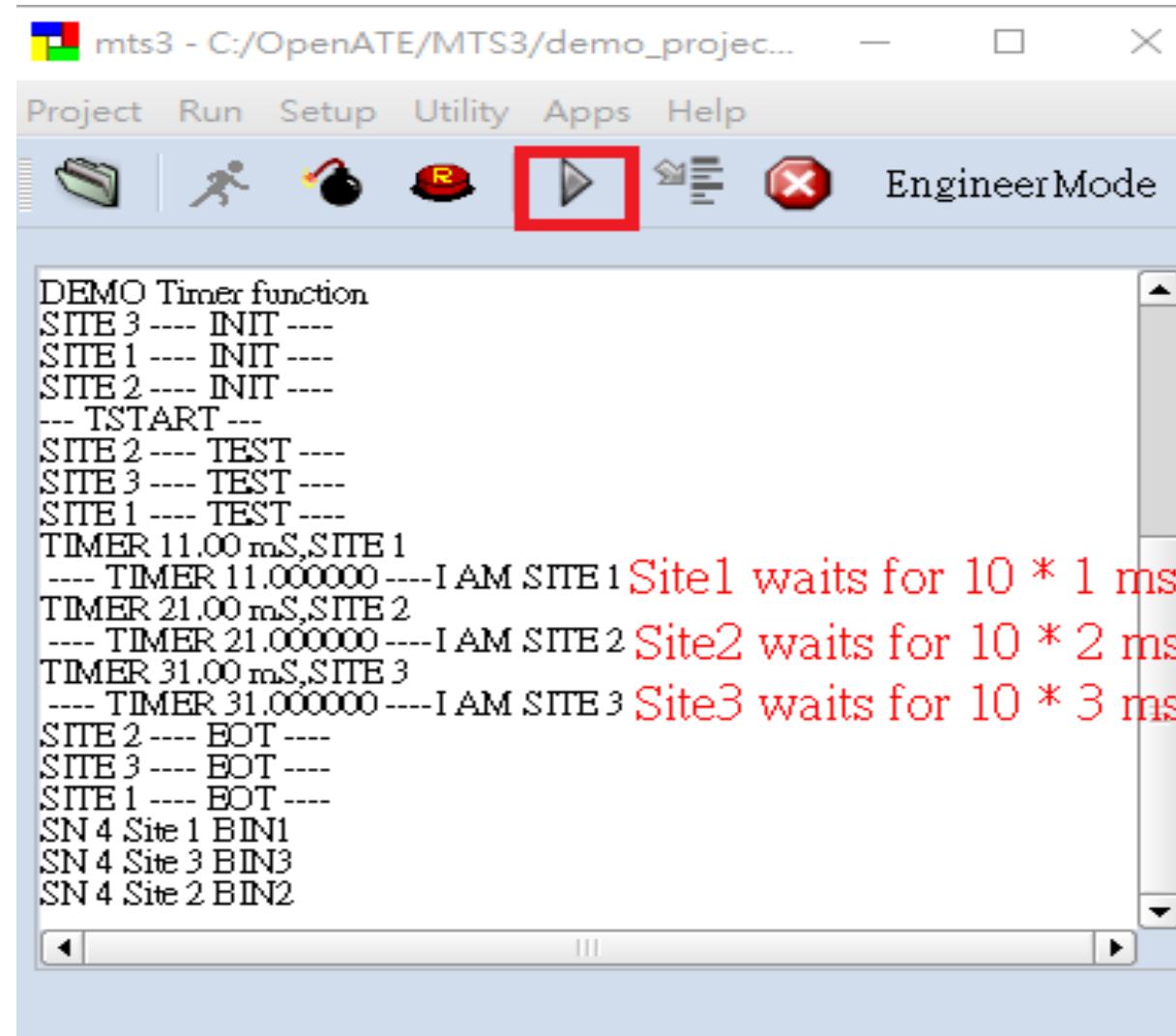


# Threading



# Step6

- Press “Tstart” button to run the test program once.



# Demo Project3

Step-by-step

# Demo Project\_3

## Demo

The screenshot displays the OpenATE MTS3 software interface. At the top, a menu bar includes GLOBAL, INIT, TEST (which is selected), EOT, PWD, and LIB. Below the menu is a code editor window containing C-like pseudocode:

```
if(mts3_get_datalog()) mts3_msg( "SITE %d ---- TEST ----", mts3_getSite() );
if(i>10) i=1;

mtS3_start_timer();

if(mts3_once()){
mtS3_msg( " ---- ONCE TEST ----I AM SITE %d", mts3_getSite() );
MSleep(100);
}
if(mts3_get_debug()) mts3_once();
mtS3_msg( " ---- TIMER %f ----I AM SITE %d", mts3_get_timer(), mts3_getSite() );
mtS3_bin(i++ );
```

Below the code editor is a project window titled "mts3 - C:/OpenATE/MTS3/demo\_projects/demo03once.mpj". It shows the following log output:

```
TIMER 100.00 mS,SITE 2
---- TIMER 100.000000 ----I AM SITE 2
SITE 1 ---- EOT ----
SITE 2 ---- EOT ----
SITE 3 ---- EOT ----
SN 3 Site 1 BIN2
SN 3 Site 3 BIN1
SN 3 Site 2 BIN3
--- TSTART ---
SITE 2 ---- TEST ---
SITE 3 ---- TEST ---
SITE 1 ---- TEST ---
---- ONCE TEST ----I AM SITE 1
TIMER 0.00 mS,SITE 2
TIMER 0.00 mS,SITE 3
---- TIMER 0.000000 ----I AM SITE 2
---- TIMER 0.000000 ----I AM SITE 3
TIMER 100.00 mS,SITE 1
---- TIMER 100.000000 ----I AM SITE 1
SITE 2 ---- EOT ----
SITE 3 ---- EOT ----
SITE 1 ---- EOT ----
SN 6 Site 3 BIN5
SN 6 Site 1 BIN6
SN 6 Site 2 BIN4
```

To the right of the project window is a control panel titled "CTL". It features a table for setting ACT (Active) and BIN values:

ACT	BIN
1 <input checked="" type="checkbox"/> Active	6
2 <input checked="" type="checkbox"/> Active	4
3 <input checked="" type="checkbox"/> Active	5

The control panel also includes a "LOOP" button set to 1000, a "Clear" button, a "Summary" button, and other control icons like a stop button and a summary report icon.

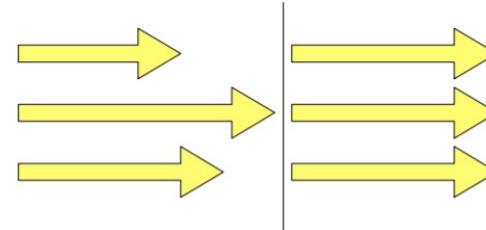
# Introduction to API

- `bool mts3_once();`
  - Sometimes for a multi-site application, a piece of code may need to be executed only once by one thread.
  - Usage:

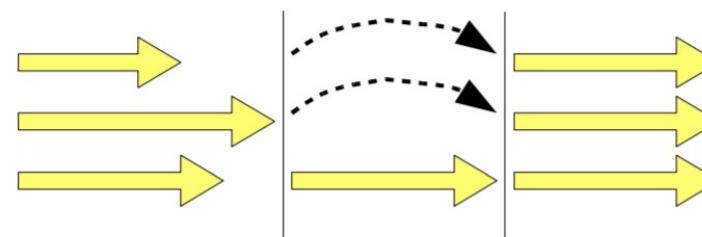
```
if ( mts3_once() ) {  
    // statements in this block will be executed only once by one thread.  
}
```
  - `mts3_once()` is also used to sync the execution among test threads.

# Synchronizing

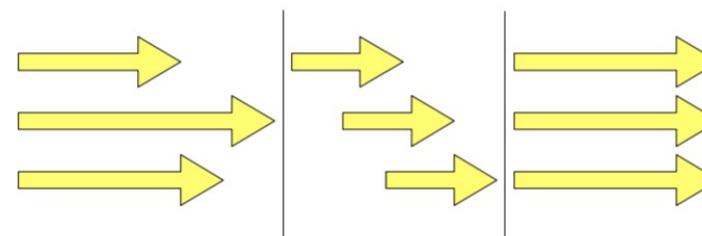
**Synch**



**Once**



**Sequence**



# Step1

- Open MTS3
- Open project “C:\OpenATE\MTS3\demo\_projects\demo03once.mpj”
- Open TPBUILD <Utility → TPBUILD >

# Step2

- You can press “Datalog” button to see more information.

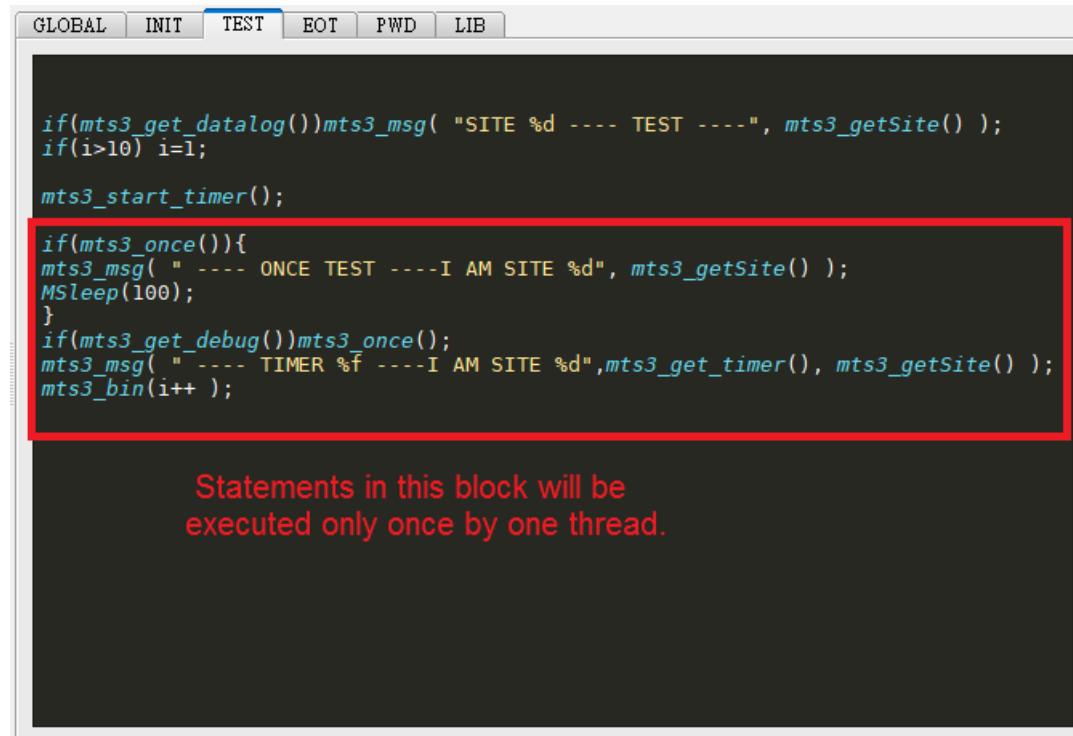
# Step3

- Increase the number of sites.
- Press “Exec” button to initialize.



# Step4

- Press “Tstart” button to run the test program once.

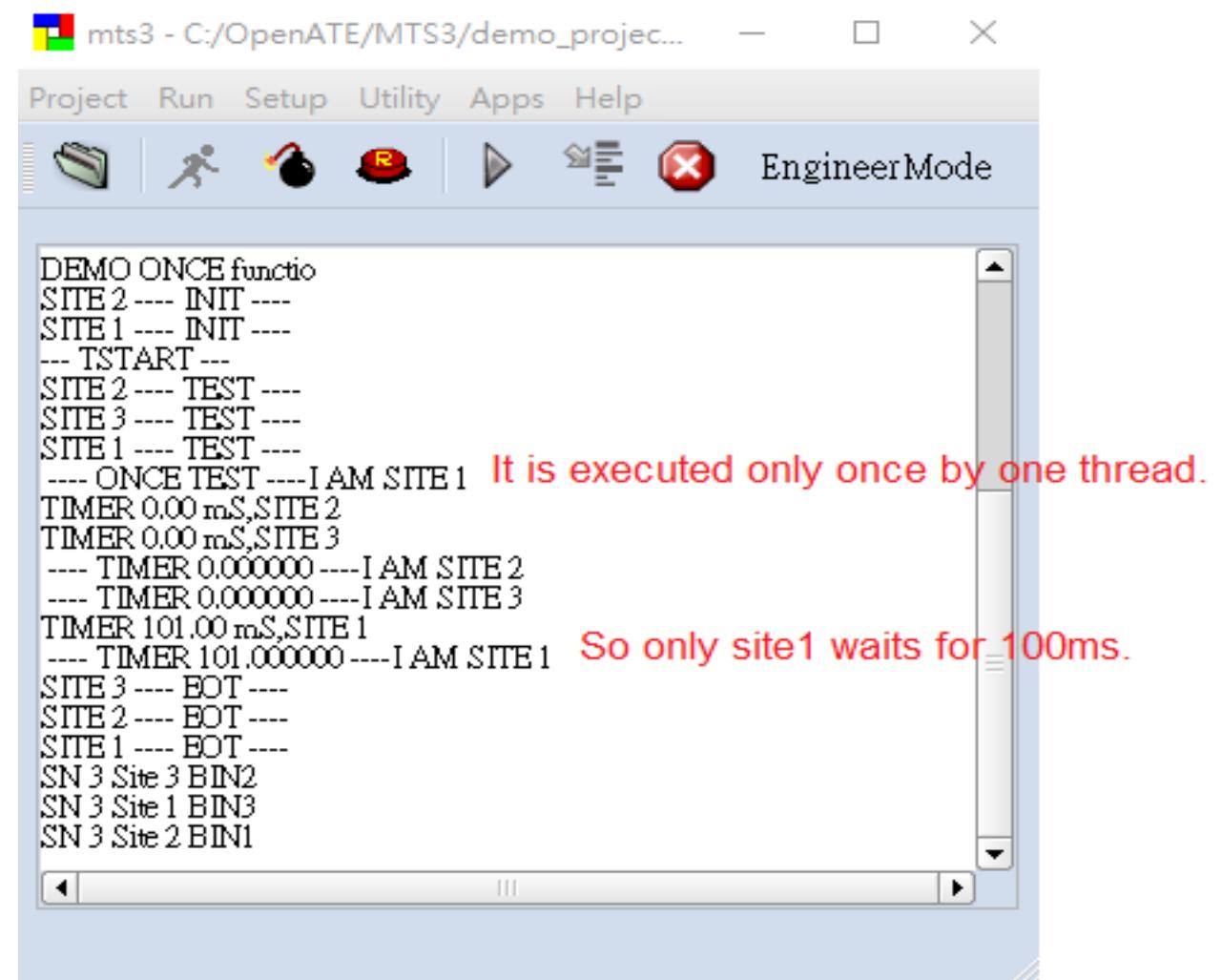


The screenshot shows the MTS3 software interface with the 'TEST' tab selected. A red box highlights the following code block:

```
if(mts3_get_datalog()) mts3_msg( "SITE %d ---- TEST ----", mts3_getSite() );
if(i>10) i=1;
mtS3_start_timer();

if(mts3_once()){
    mts3_msg( " ---- ONCE TEST ---- I AM SITE %d", mts3_getSite() );
    MSleep(100);
}
if(mts3_get_debug()) mts3_once();
mtS3_msg( " ---- TIMER %f ---- I AM SITE %d", mts3_get_timer(), mts3_getSite() );
mtS3_bin(i++);
```

Statements in this block will be executed only once by one thread.



The screenshot shows the MTS3 software interface with the 'TEST' tab selected. The output window displays the following log entries:

```
DEMO ONCE function
SITE 2 ---- INIT ----
SITE 1 ---- INIT ----
--- TSTART ---
SITE 2 ---- TEST ----
SITE 3 ---- TEST ----
SITE 1 ---- TEST ----
---- ONCE TEST ---- I AM SITE 1 It is executed only once by one thread.
TIMER 0.00 mS, SITE 2
TIMER 0.00 mS, SITE 3
---- TIMER 0.000000 ---- I AM SITE 2
---- TIMER 0.000000 ---- I AM SITE 3
TIMER 101.00 mS, SITE 1
---- TIMER 101.000000 ---- I AM SITE 1 So only site1 waits for 100ms.
SITE 3 ---- EOT ----
SITE 2 ---- EOT ----
SITE 1 ---- EOT ----
SN 3 Site 3 BIN2
SN 3 Site 1 BIN3
SN 3 Site 2 BIN1
```

# Demo Project4

Step-by-step

# Demo Project\_4

## Demo

The screenshot shows the OpenATE software interface during a test sequence. The top menu bar includes GLOBAL, INIT, TEST (selected), EOT, PWD, and LIB. The code editor window displays C-like pseudocode for a test sequence involving three sites (SITE 1, SITE 2, SITE 3). The sequence involves sending messages like "NO SEQ" and "WITH SEQ" followed by various test commands (SYN TEST 1-8) and status checks ("I AM SITE").

A control panel window at the bottom left shows a table with three rows labeled 1, 2, and 3, each with an "Active" checkbox checked. Buttons for LOOP, 1000ms, Clear, Summary, MAX SITE (set to 3), and RUN TP are also present.

The main log window on the right shows the execution of the test sequence. It is divided into two sections by a red box:

- Before using SEQ function:** Logs show initial site initialization and a sequence of SYN TEST commands from all three sites.
- After using SEQ function:** Logs show the sequence "WITH SEQ ---I AM SITE 1" followed by a sequence of SYN TEST commands from Site 1 only, indicating the scope of the SEQ function.

Below the log window, additional log entries show EOT (End Of Test) messages and binning information for Site 3.

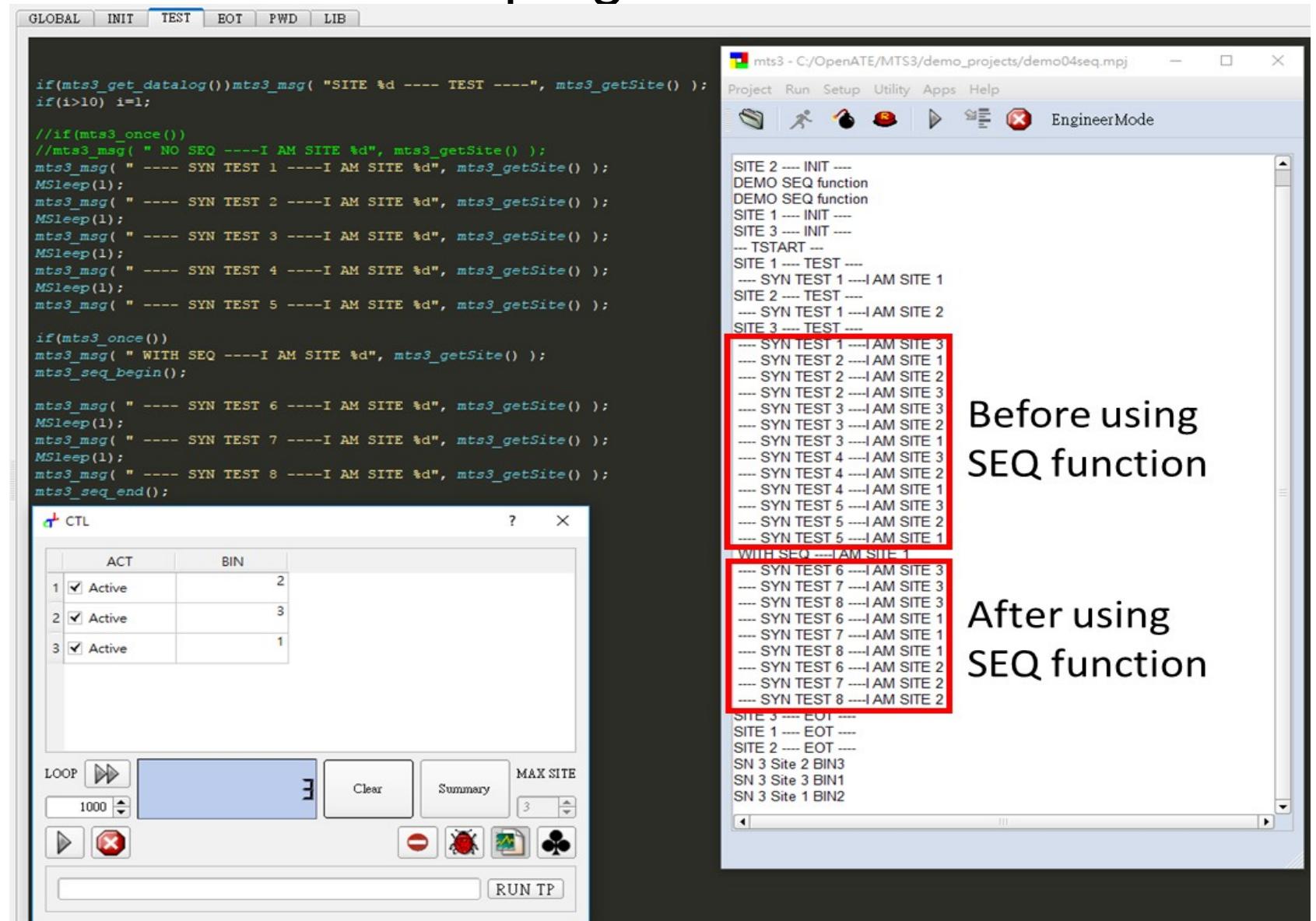
# Introduction to API

- `void mts3_seq_begin()` / `void mts3_seq_end()`;
  - Sometimes for a multi-site application, a piece of code may not able to be executed concurrently by more than one thread at the same time.
  - Usage:

```
mts3_seq_begin(); // ---- Point A  
// statements in this block will be executed by one thread at a time.  
mts3_seq_end(); // ---- Point B
```

# Step4

- Press “Tstart” button to run the test program once.



# Demo Project5

Step-by-step

# Demo Project 5

## Demo

The screenshot shows the OpenATE MTS3 software interface with three main windows:

- Code Editor (TEST tab):** Displays C code for a test sequence. The code includes logic to check if a datalog is present, send a message, and enter a loop where it pushes a stop command to break an infinite loop. It also increments a counter and sends a bin message.
- Terminal Window:** Shows the output of the test sequence:

```
-- TSTART --
SITE 1 --- TEST ---
SITE 1 push STOP TO BREAK infinite loop ---
-- STOP --
SITE 1 --- EOT ---
```
- CTL (Control) Window:** A graphical user interface for controlling the test sequence. It features a table with columns ACT and BIN, showing row 1 with Active checked and BIN 0. Below the table are buttons for LOOP (set to 1000), Clear, Summary, MAX SITE (set to 1), and RUN TP.

# Introduction to API

- `void mts3_check_stop();`
  - In user's test program, add this function call in a infinite loop, so that system can interrupt the loop when user click 'stop'.
    - `while (1) { if ( hardware_ready() ) exit; }`
- Here we put a call to `mts3_check_stop()` in the loop, so that user can hit stop button to stop the infinite loop:
  - `while (1) {  
 if ( hardware_ready() ) exit;  
 mts3_check_stop();}`

# Step1

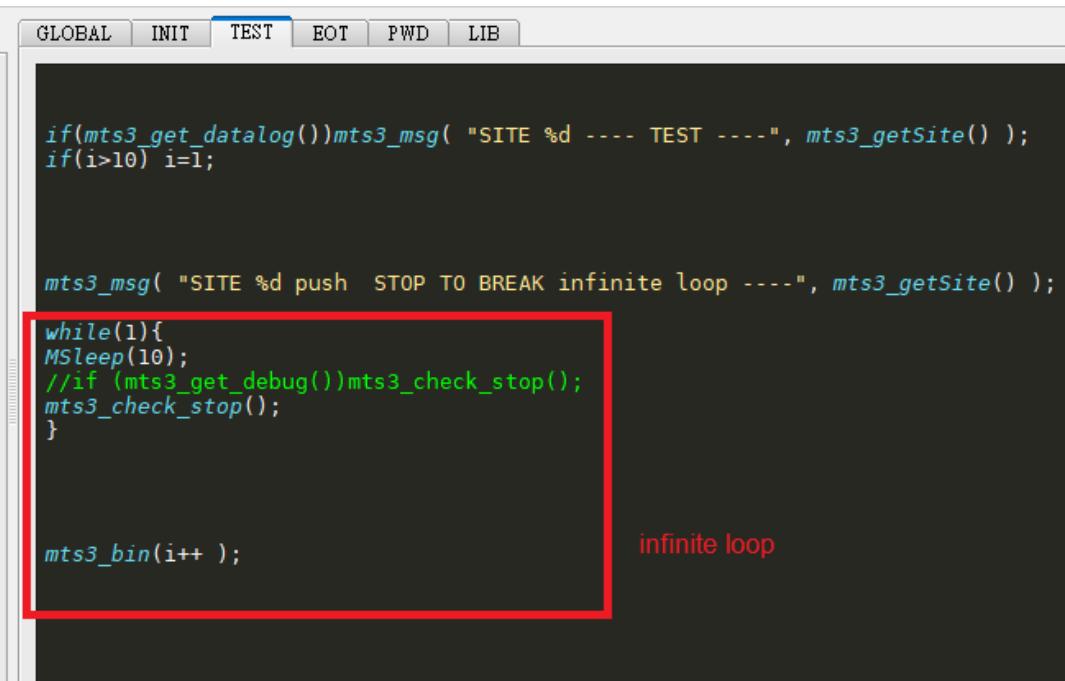
- Open MTS3

Open project “C:\OpenATE\MTS3\demo\_projects\demo05chkstop.mpj”

- Open TPBUILD <Utility → TPBUILD >

# Step2

- Press “Exec” button to initialize.
- Press “Tstart” button to run the test program once.



The screenshot shows the TPBUILD software interface with a menu bar at the top containing GLOBAL, INIT, TEST (which is highlighted), EOT, PWD, and LIB. The main window displays a block of C code. A red rectangular box highlights a section of the code, and the text "infinite loop" is written in red to its right. The code is as follows:

```
if(mts3_get_datalog()) mts3_msg( "SITE %d ---- TEST ----", mts3_getSite() );
if(i>10) i=1;

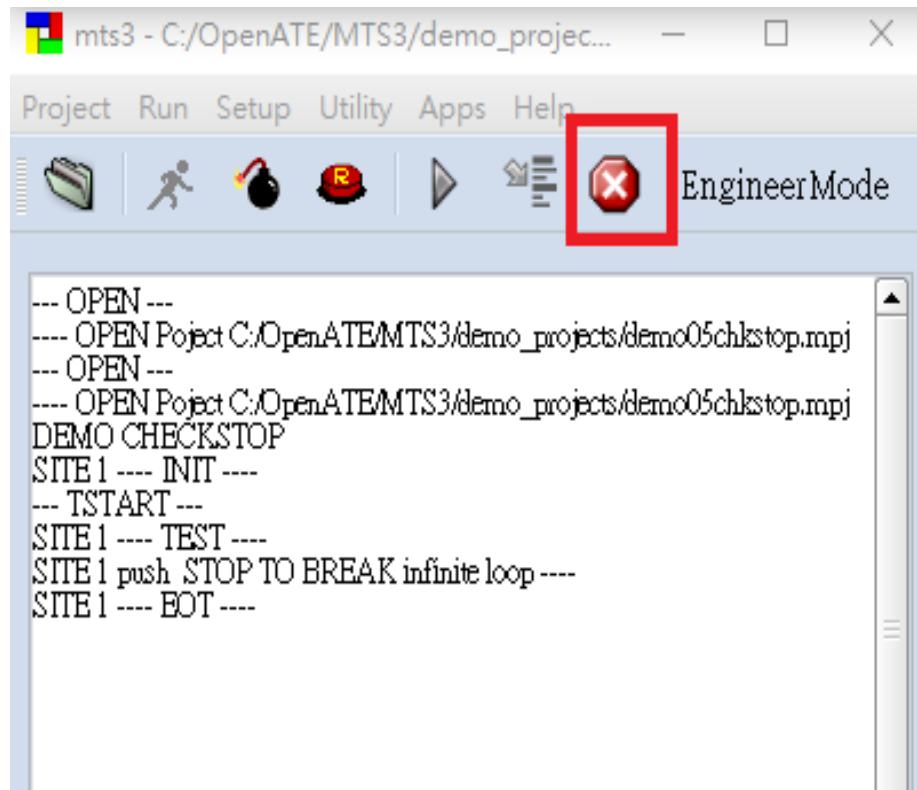
mts3_msg( "SITE %d push STOP TO BREAK infinite loop ----", mts3_getSite() );

while(1){
    MSleep(10);
    //if (mts3_get_debug()) mts3_check_stop();
    mts3_check_stop();
}

mts3_bin(i++ );
```

# Step3

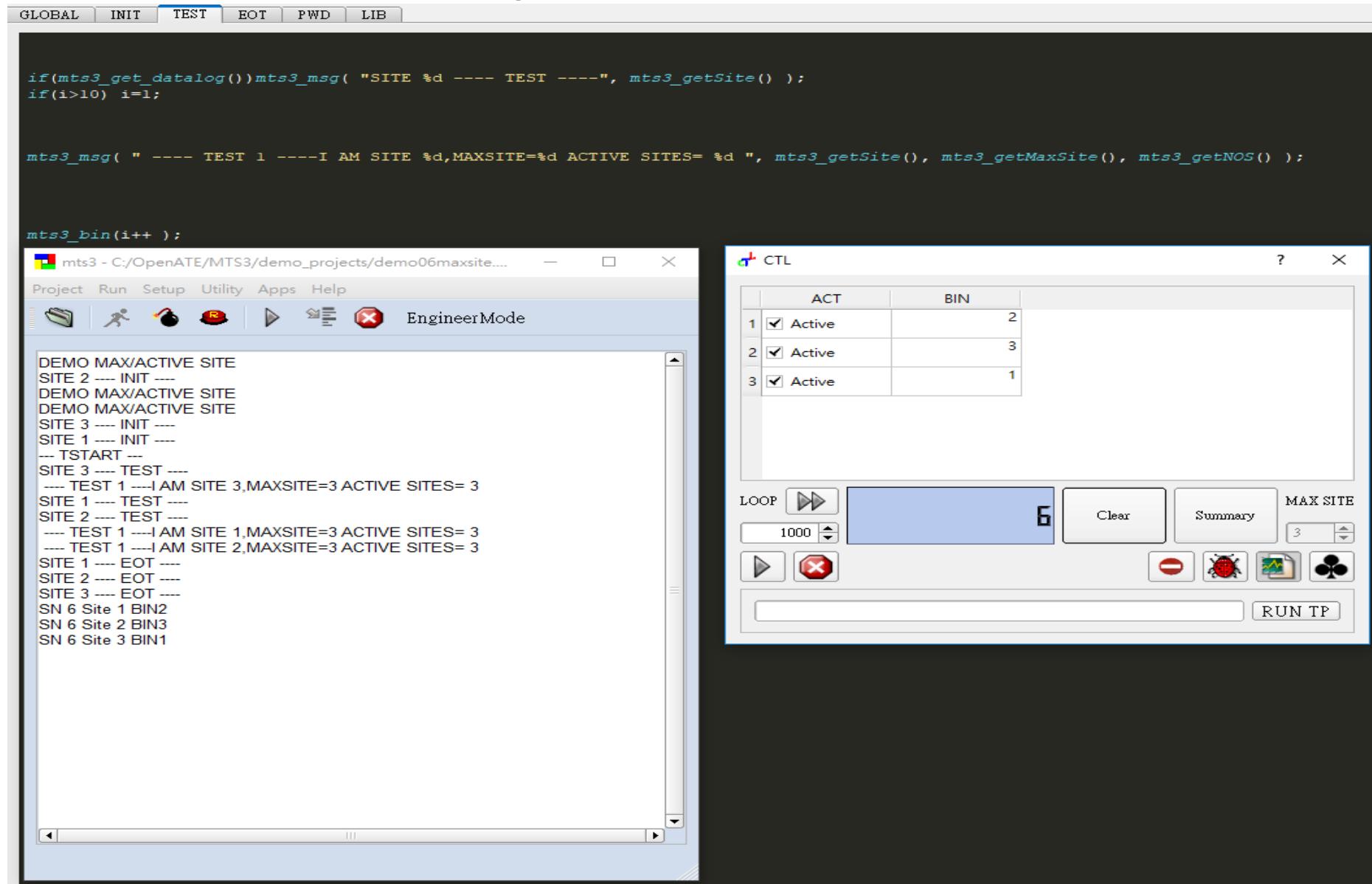
- Push STOP to break infinite loop



# Demo Project6

# Demo Project\_6

- Setup
  - Three sites
- Max site = 3
- Active site = 3



# Demo Project 6

- Setup
  - Three sites
  - Deactivate third site
- Max site = 3
- Active site = 2

The screenshot displays the OpenATE MTS3 software interface with three main windows:

- Code Editor:** Shows C code for the demo project. It includes logic to check if a site is active and to print test messages.
- Terminal:** Shows the terminal output from the device. The log includes initialization messages for Site 2, Site 3, and Site 1, followed by test messages indicating Site 3 is active (MAXSITE=3, ACTIVE SITES= 3), and Site 1 and Site 2 being active (MAXSITE=3, ACTIVE SITES= 2). It also shows EOT (End Of Test) messages and site binning information (SN 6 Site 1 BIN2, SN 6 Site 2 BIN3, SN 6 Site 3 BIN1).
- Control Panel:** Shows the configuration for the active sites. A table lists sites 1 and 2 as active (ACT checked, BIN values 4 and 5 respectively), while site 3 is inactive (ACT unchecked, BIN value 1). The control panel also includes a loop counter (set to 1000), a start/stop button, and various status indicators.

```
if(mts3_get_datalog()) mts3_msg( "SITE %d ---- TEST ----", mts3_getSite() );
if(i>10) i=1;

mts3_msg( " ---- TEST 1 ----I AM SITE %d,MAXSITE=%d ACTIVE SITES= %d ", mts3_getSite(), mts3_getMaxSite(), mts3_getNOS() );

mts3_bin(i++ );
```

```
DEMO MAX/ACTIVE SITE
SITE 2 --- INIT ---
DEMO MAX/ACTIVE SITE
DEMO MAX/ACTIVE SITE
SITE 3 --- INIT ---
SITE 1 --- INIT ---
-- TSTART --
SITE 3 --- TEST ---
--- TEST 1 ---I AM SITE 3,MAXSITE=3 ACTIVE SITES= 3
SITE 1 --- TEST ---
SITE 2 --- TEST ---
--- TEST 1 ---I AM SITE 1,MAXSITE=3 ACTIVE SITES= 3
--- TEST 1 ---I AM SITE 2,MAXSITE=3 ACTIVE SITES= 3
SITE 1 --- EOT ---
SITE 2 --- EOT ---
SITE 3 --- EOT ---
SN 6 Site 1 BIN2
SN 6 Site 2 BIN3
SN 6 Site 3 BIN1
-- TSTART --
SITE 1 --- TEST ---
--- TEST 1 ---I AM SITE 1,MAXSITE=3 ACTIVE SITES= 2
SITE 2 --- TEST ---
--- TEST 1 ---I AM SITE 2,MAXSITE=3 ACTIVE SITES= 2
SITE 1 --- EOT ---
SITE 2 --- EOT ---
SN 8 Site 1 BIN4
SN 8 Site 2 BIN5
```

ACT	BIN
<input checked="" type="checkbox"/> Active	4
<input checked="" type="checkbox"/> Active	5
<input type="checkbox"/> Active	1

CTL

LOOP: 1000

Buttons: Clear, Summary, MAX SITE, RUN TP, Stop, Status

# Introduction to API

- `int mts3_getMaxSite();`
  - Return maximum sites available by system.
- `int mts3_getNOS();`
  - Return number of sites for current test configuration.

# Reference

- OpenATE official web
  - <http://www.openate.com/>
- mts3
- mts3\_user.manual