

# MTS3

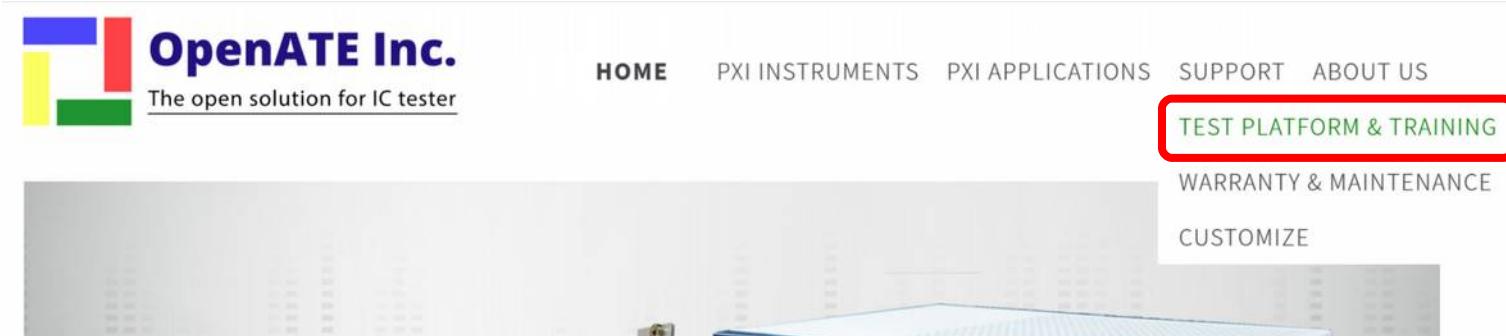
# Environment Setup

# Preparing for using MTS3

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

# 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.



### DOWNLOAD

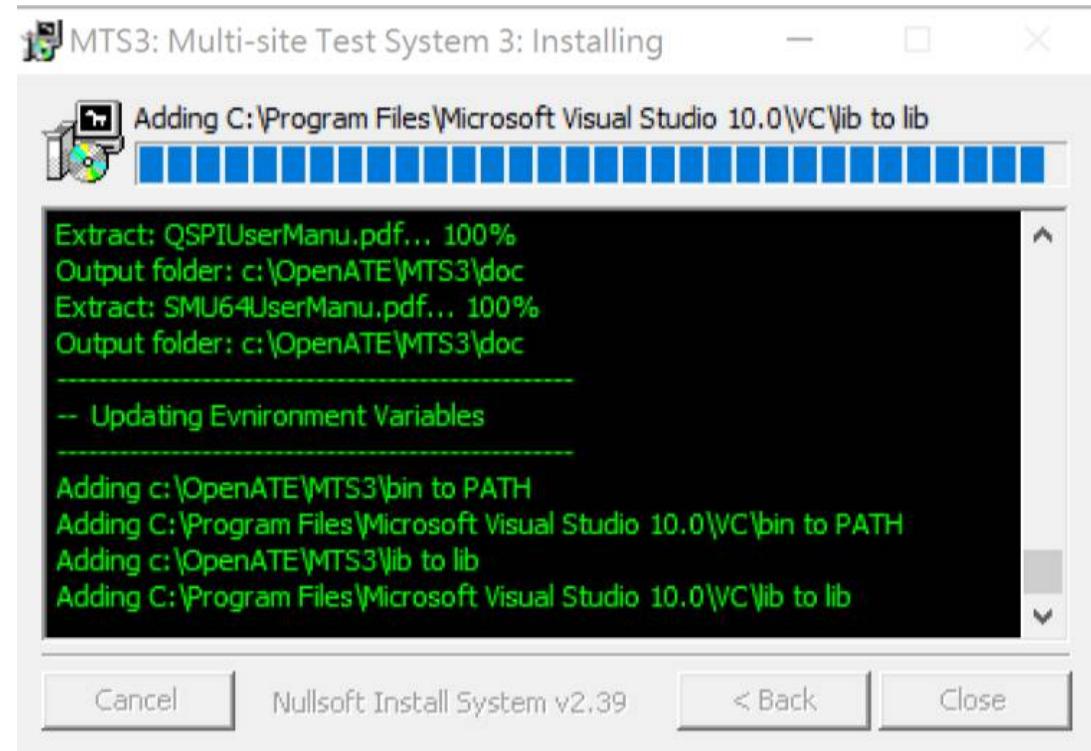
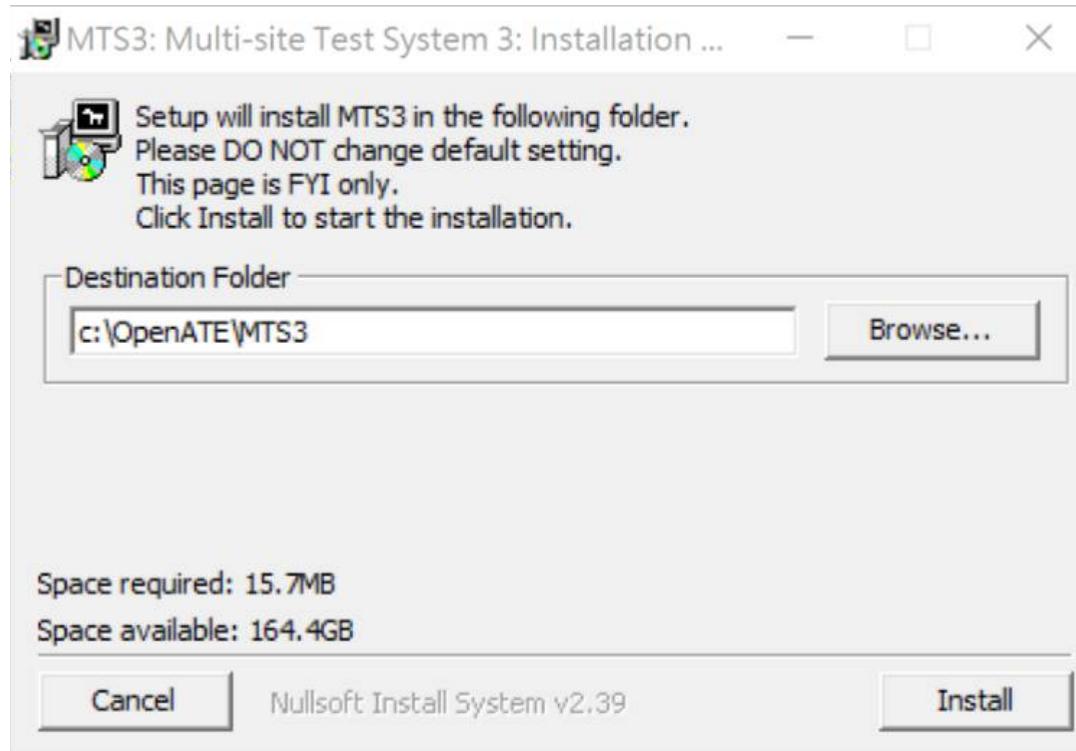
- 1 [Data sheet](#)
- 2 [Install](#)

[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\_VISAvs13install20191230.exe



- Step 2:

Install Compiler “**Visual C++ 2013 Express**”

Complete the following instructions after installing the VC++:

- (a) Add “**C:\Program Files\Microsoft Visual Studio 12.0\VC\bin**”  
“**C:\Program Files\Microsoft Visual Studio 12.0\Common7\IDE**”  
to **PATH** in Environment Variable
- (b) Add “**C:\Program Files (x86)\Microsoft Visual Studio 12.0\VC\include**”  
to **include** in Environment Variable
- (c) Add “**C:\Program Files (x86)\Microsoft Visual Studio 12.0\VC\lib**”  
to **lib** in Environment Variable
- (d) Add “C:\Program Files (x86)\Adobe\Reader11.0\Reader” to **PATH**.

- Step 3: Download “**NI-VISA**” and install

# Introduction to MTS3

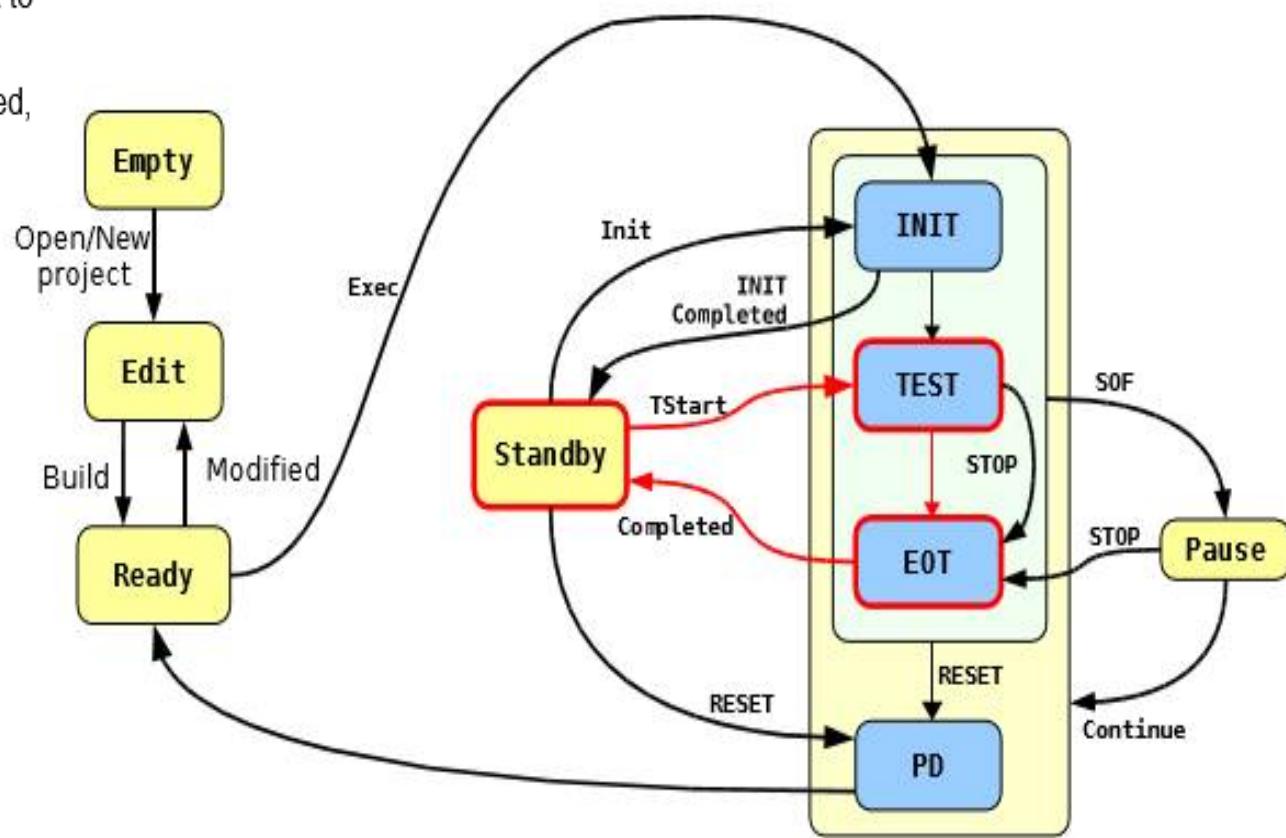
# Documents

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

# 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.



# User Interface for MTS3

Open Project

Exec

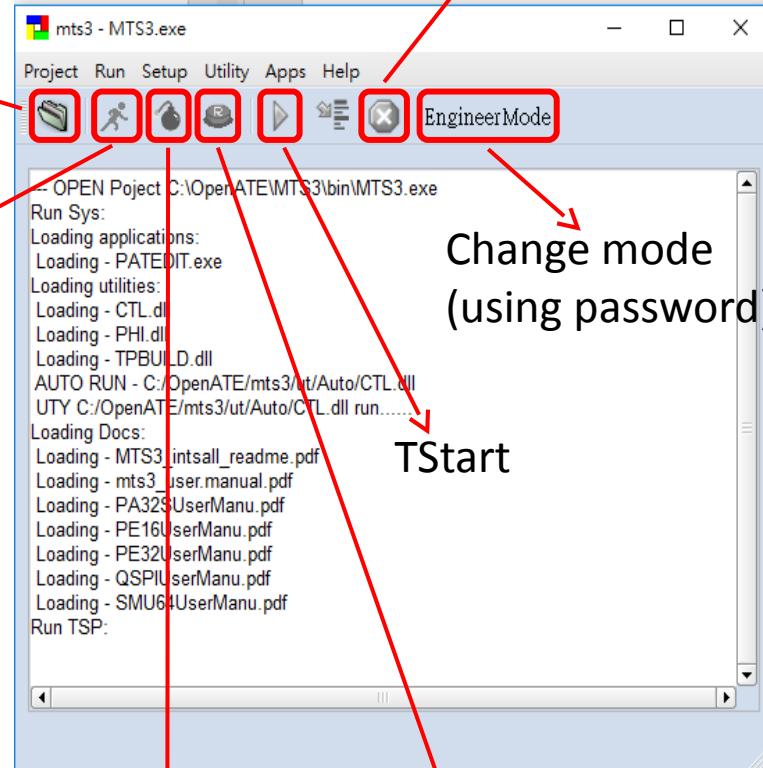
Init

Reset

Stop

Change mode  
(using password)

TStart



Stop on fail

Debug

Show the site

Determine the number of loop

LOOP

1000

MAX SITE

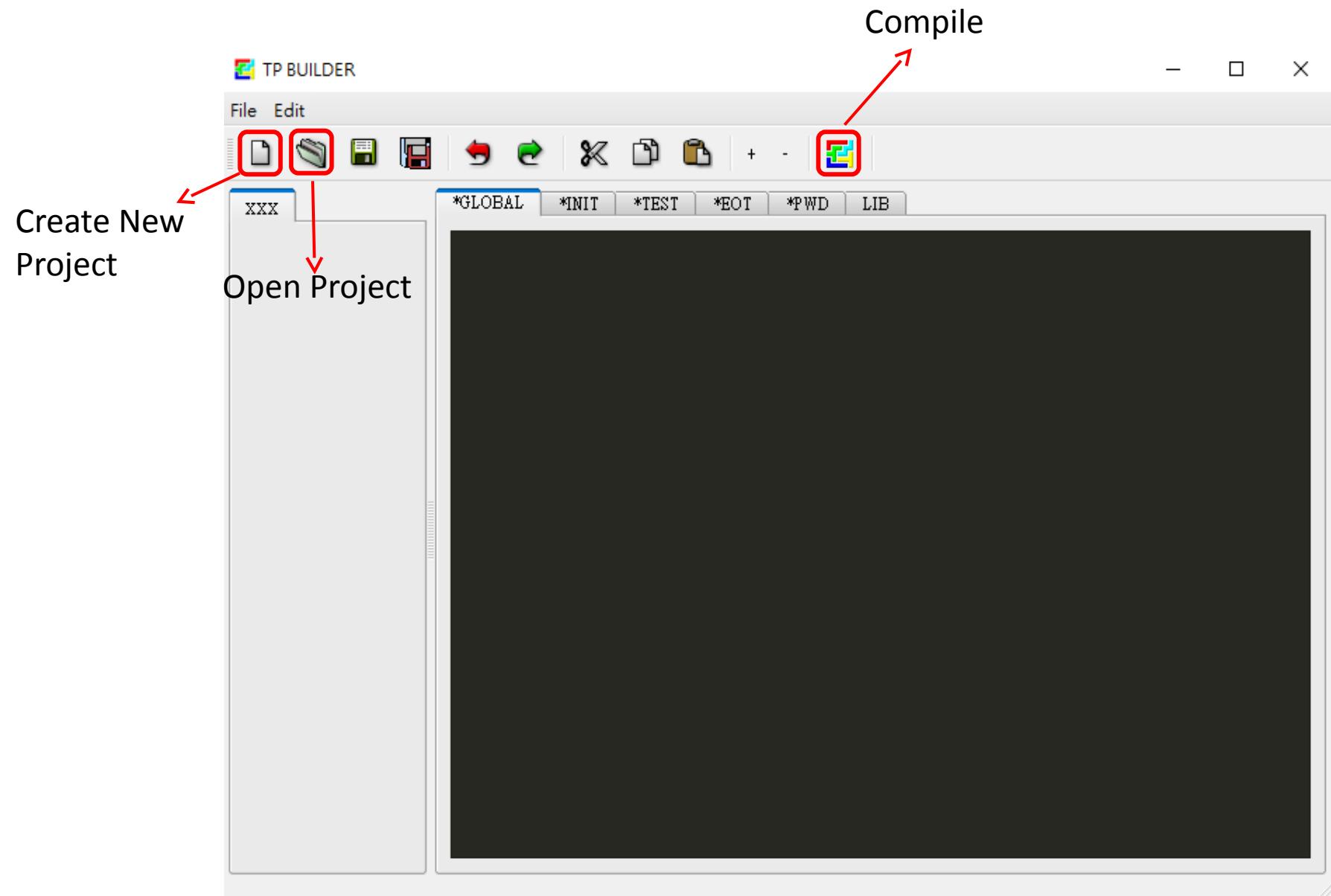
1

Determine the  
number of site

Override

Datalog

- Utility → TPBUILD



# GLOBAL

- Declare global variables and functions.

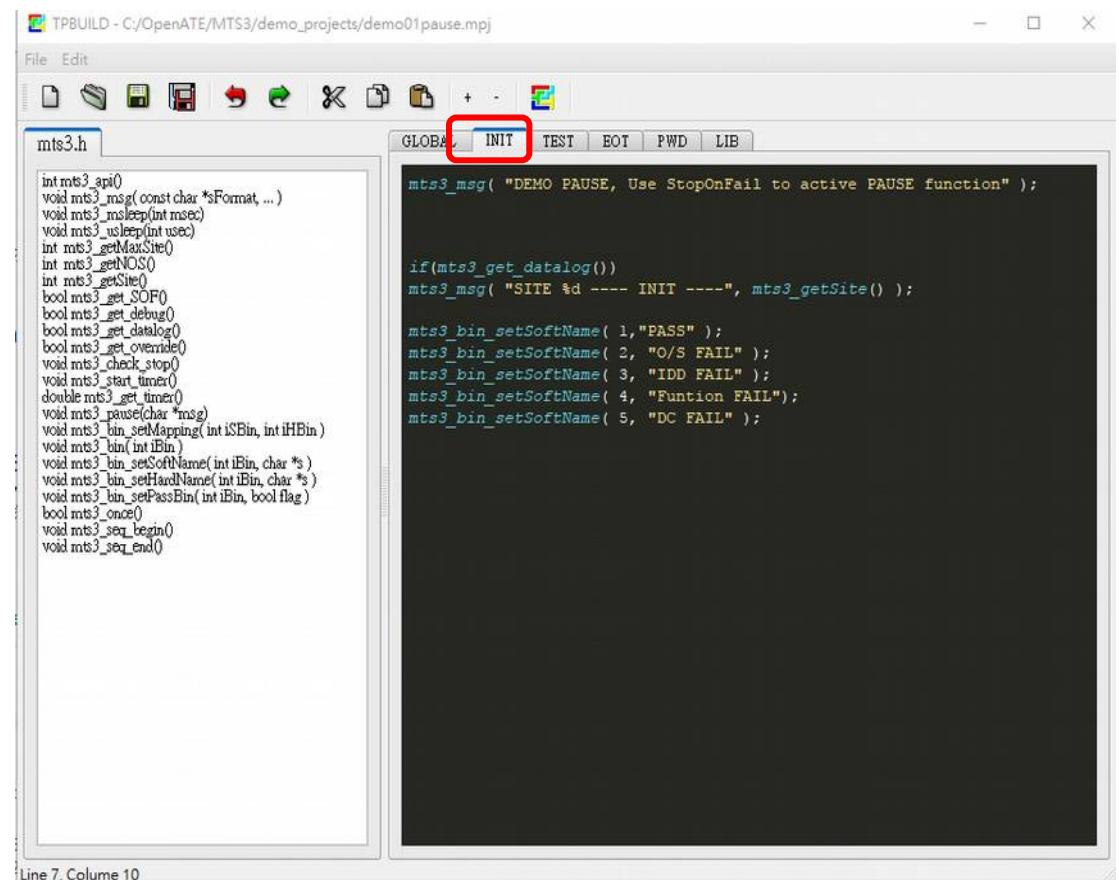
The screenshot shows a software interface titled "TPBUILD - C:/OpenATE/MTS3/demo\_projects/demo01pause.mpj". The window has a toolbar with various icons at the top. Below the toolbar, there are several tabs: GLOBAL, INIT, TEST, EOT, PWD, and LIB. The GLOBAL tab is currently selected and highlighted with a red box. To the left of the tabs, there is a code editor window with a title bar "mts3.h". The code in the editor is as follows:

```
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()
```

At the bottom of the code editor, it 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.



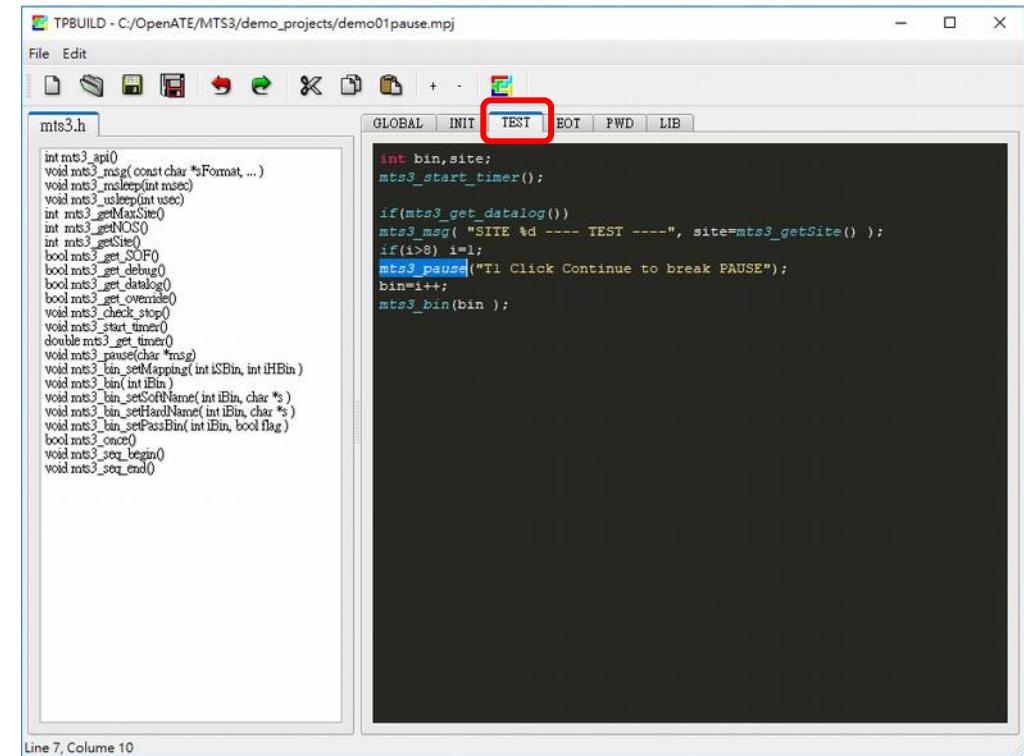
The screenshot shows the TPBUILD software interface with the file "mts3.h" open. The tabs at the top are GLOB, INIT (which is highlighted with a red box), TEST, EOT, PWD, and LIB. The code in the editor window is as follows:

```
TPBUILD - C:/OpenATE/MTS3/demo_projects/demo01pause.mpj
File Edit
mts3.h
GLOB 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" );
int mts3_api()
void mts3_msg(const char *sFormat, ... )
void mts3_msleep(int msec)
void mts3_usleep(int usec)
int mts3_getMaxSite()
int mts3_getNOSO
int mts3_getSite()
bool mts3_get_SOF0
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()
void mts3_seq_end()
```

Line 7, Column 10

# 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 title bar "TPBUILD - C:/OpenATE/MTS3/demo\_projects/demo01pause.mpj". Below the title bar is a toolbar with various icons. To the right of the toolbar is a menu bar with "File", "Edit", and other options. The main area is divided into two panes. The left pane displays the source code file "mts3.h" with several function declarations. The right pane shows the "TEST" tab selected, containing a block of C-like pseudocode. A red box highlights the "TEST" tab in the tab bar. The status bar at the bottom indicates "Line 7, Column 10".

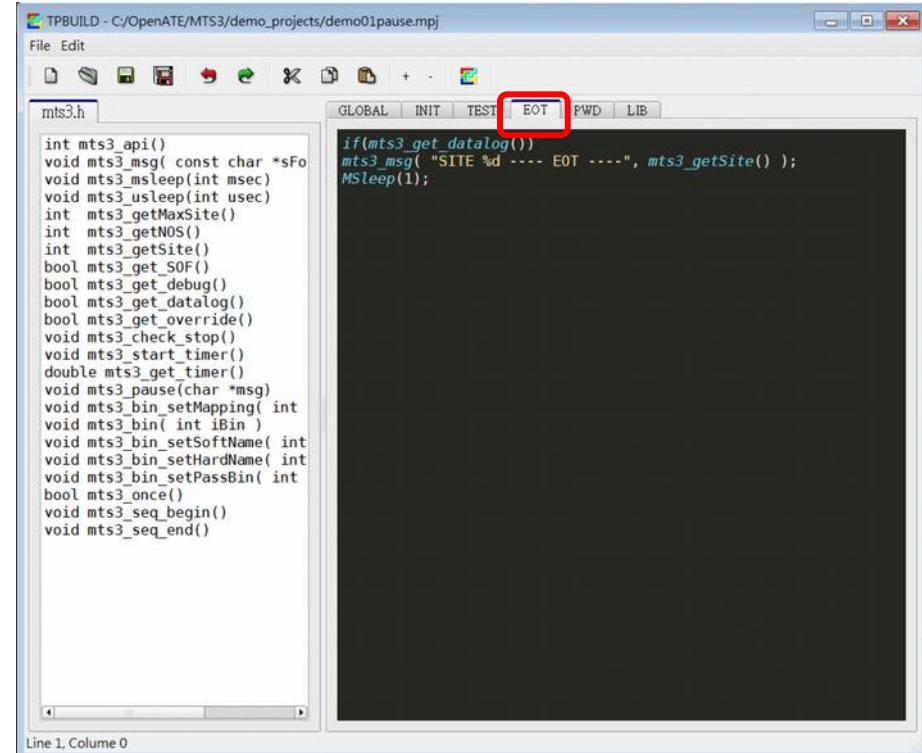
```
int mts3_api0
void mts3_msg(const char *Format, ... )
void mts3_msleep(int msec)
void mts3_usleep(int usec)
int mts3_getMaxSite()
int mts3_getNOS()
int mts3_getSite()
bool mts3_get_SOFO()
bool mts3_get_debug()
bool mts3_get_datalog()
bool mts3_get_overidle()
void mts3_check_stop()
void mts3_start_timer()
double mts3_get_timer()
void mts3_pause(char *msg)
void mts3_bin_setMapping( int iBin, int iIBin )
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_set_begin()
void mts3_set_end()
```

```
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("Tl Click Continue to break PAUSE");
bin++;
mtS3_bin(bin );
```

# EOT

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



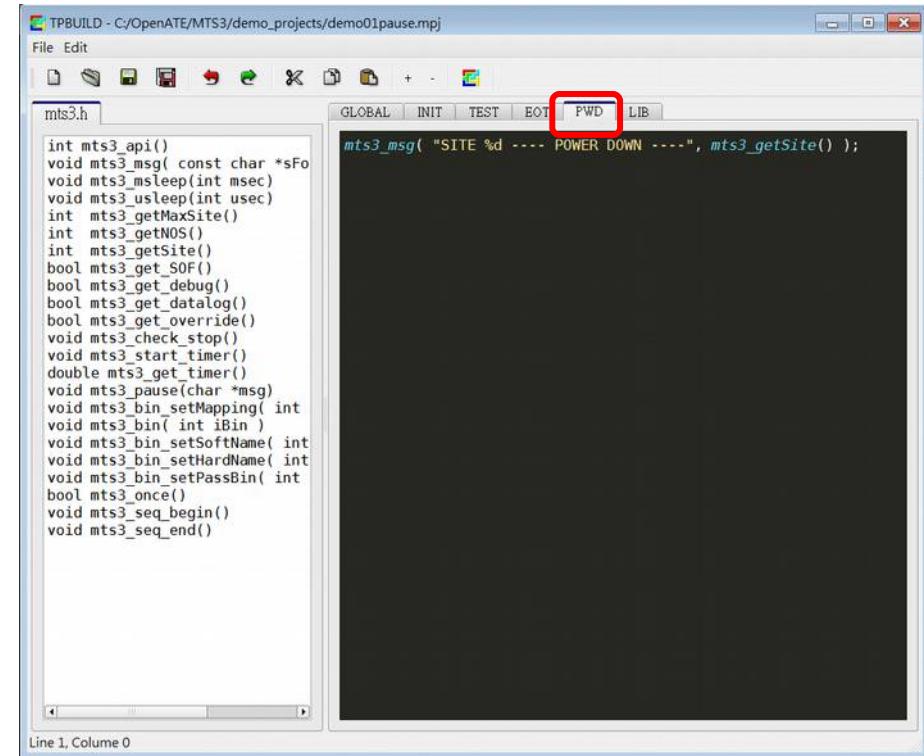
The screenshot shows a software interface titled "TPBUILD - C:/OpenATE/MTS3/demo\_projects/demo01pause.mpj". The window has tabs at the top: GLOBAL, INIT, TEST, EOT, PWD, and LIB. The EOT tab is highlighted with a red box. On the left, there is a code editor window titled "mts3.h" containing a list of C functions. On the right, there is a large black text area where code is being written or displayed. The code in the text area includes an if statement that checks if the mts3\_get\_datalog() function is called, and if so, it prints a message to the serial port and sleeps for 1 second.

```
int mts3_api()
void mts3_msg( const char *sFo
void mts3_msleep(int msec)
void mts3_usleep(int usec)
int mts3_getMaxSite()
int mts3_getNOSI()
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.



```
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()

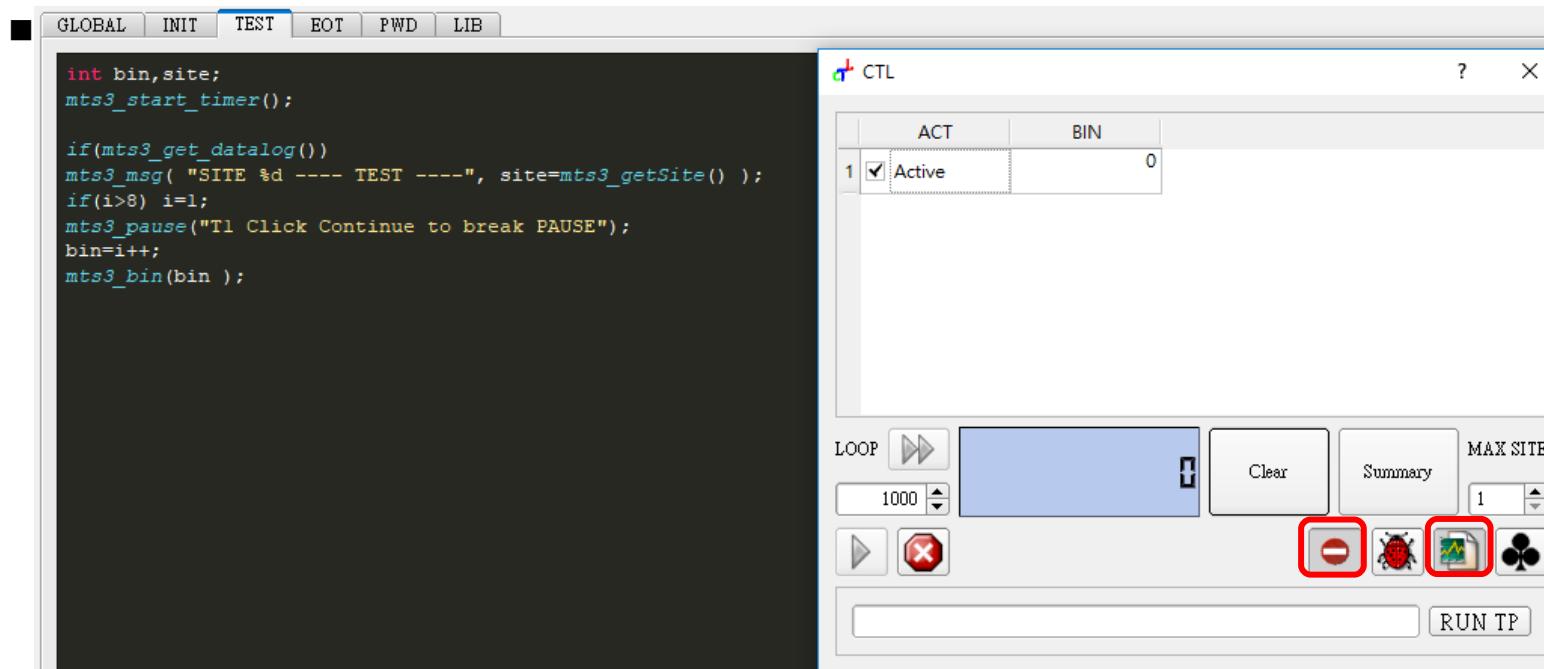
mts3_msg( "SITE %d ---- POWER DOWN ----", mts3_getSite() );
```

# Demo Project1

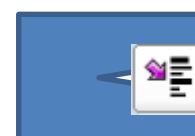
Step-by-step

# Demo Project\_1

## Demo PAUSE



PAUSE and click continue



# Introduction to API

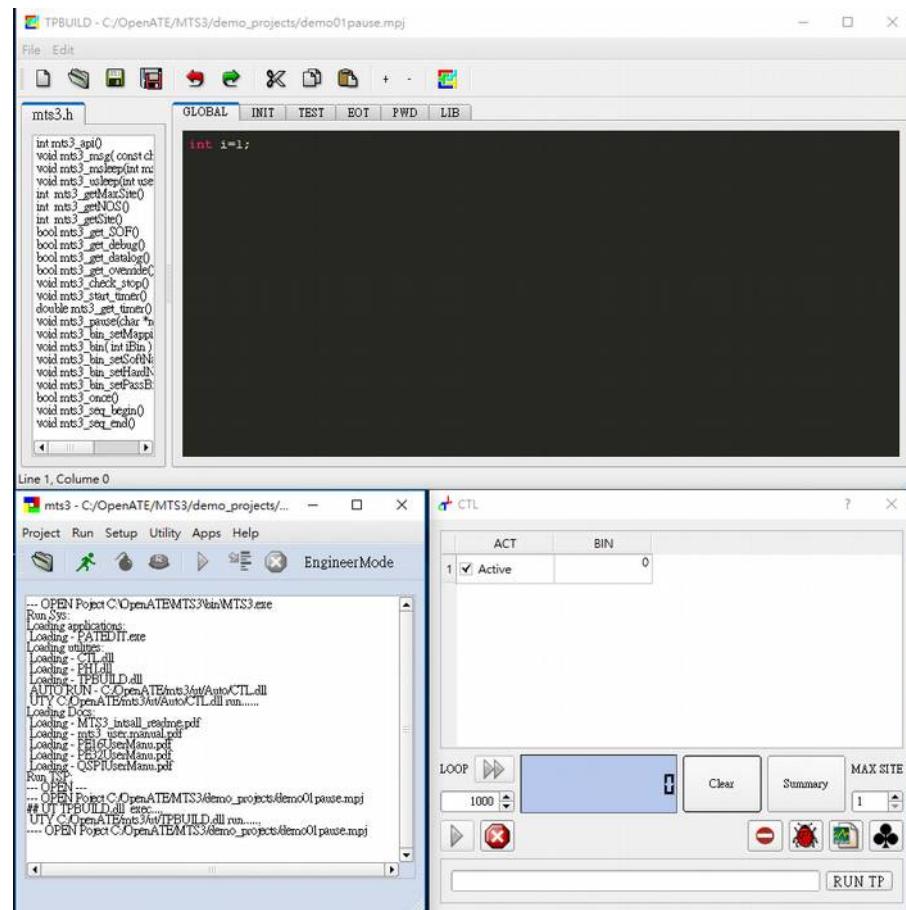
- `void mts3_msg( const char *sFormat, ... );`
  - Print message.
  - Usage: arguments are same as c standard function “printf”.
- `bool mts3_get_datalog();`
  - Return button 'Datalog' state of main dialog.
- `void mts3_bin_setSoftName( int iBin, char *s );`
  - Set name of specified software bin.

# 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

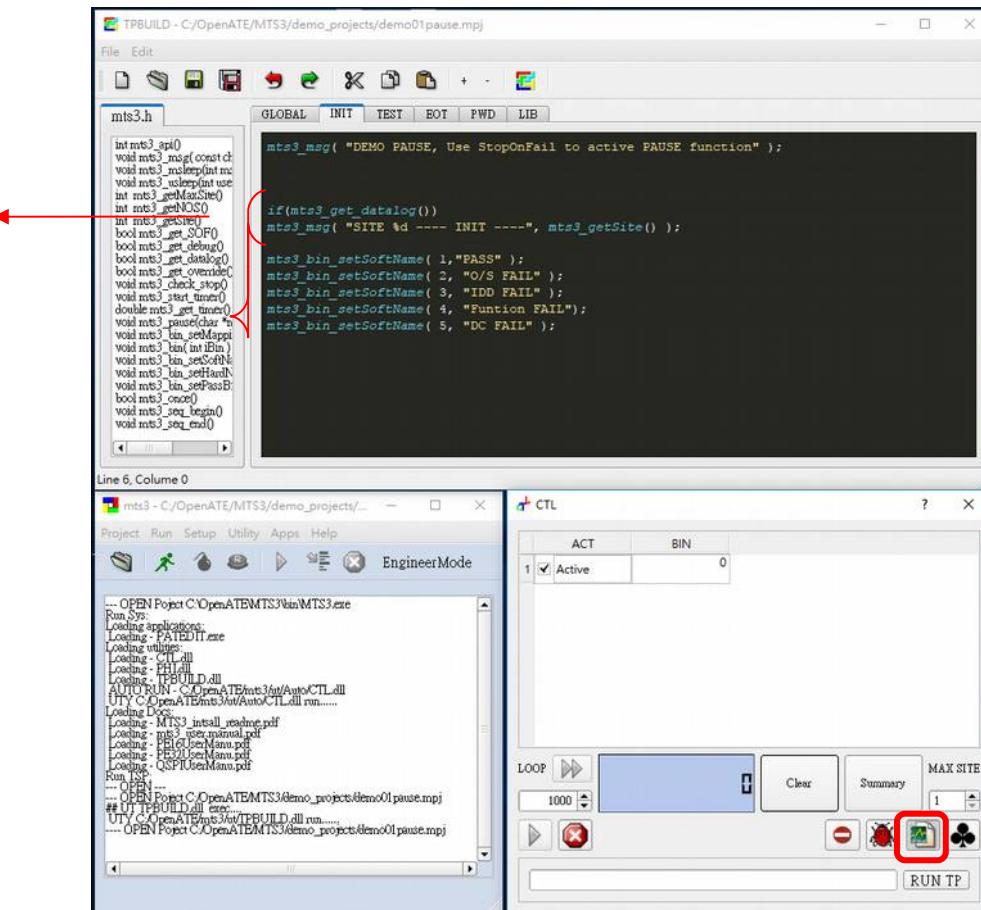
- Open MTS3
- Open project “C:\OpenATE\MTS3\demo\_projects\demo01pause.mpj”
- Open TPBUILD
  - Utility → TPBUILD



# Step2

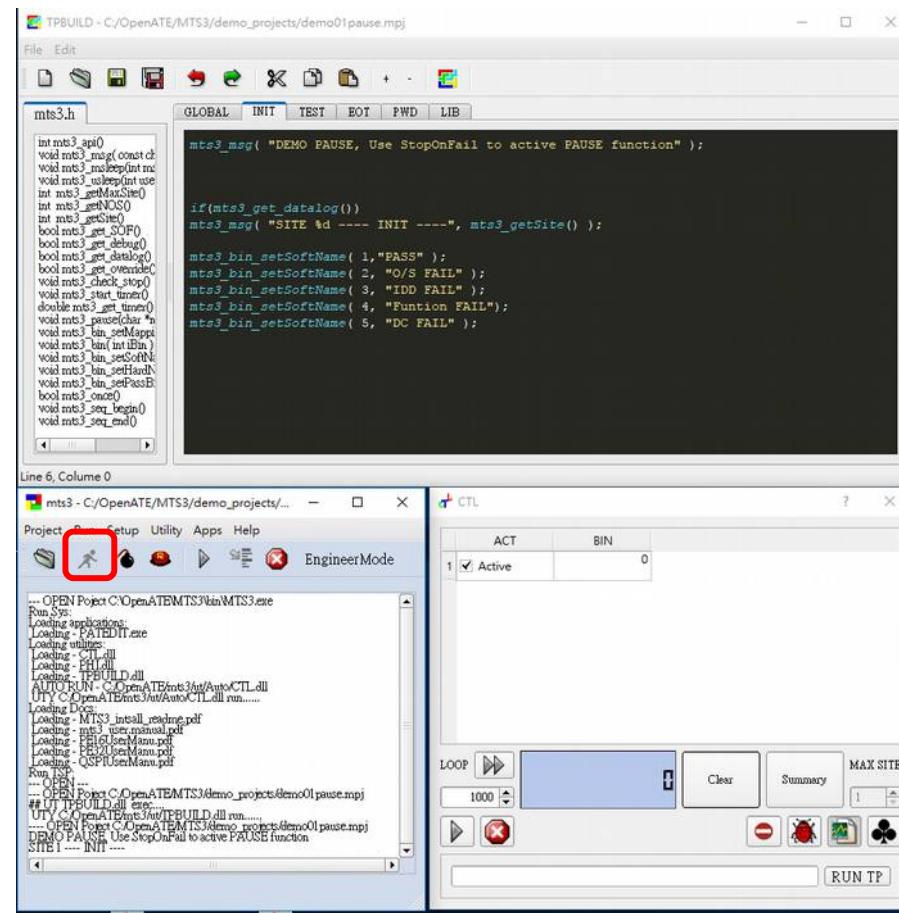
- You can press “Datalog” button to see some information.

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



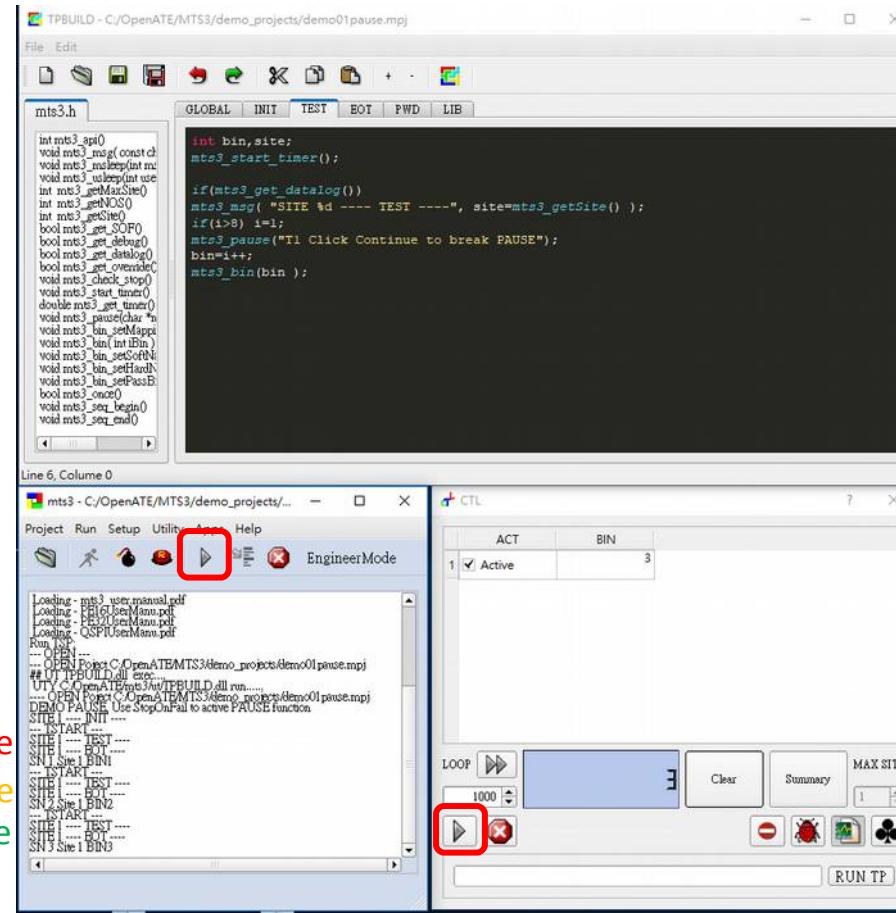
# Step3

- Press “Exec” button to initialize.



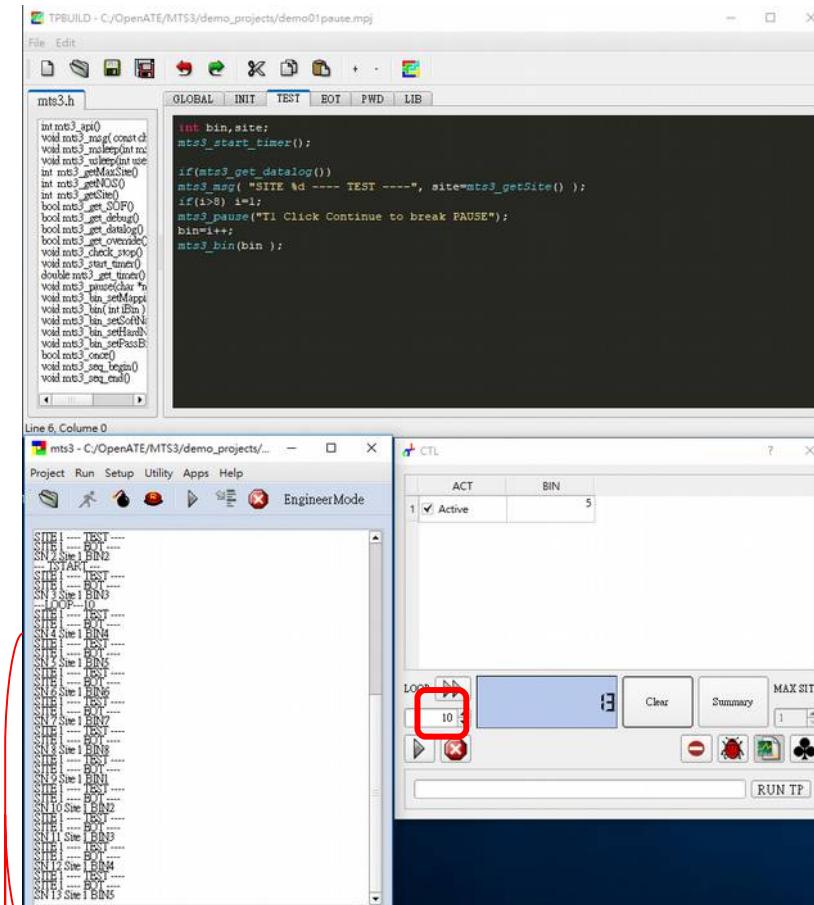
# Step4

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



# Step5

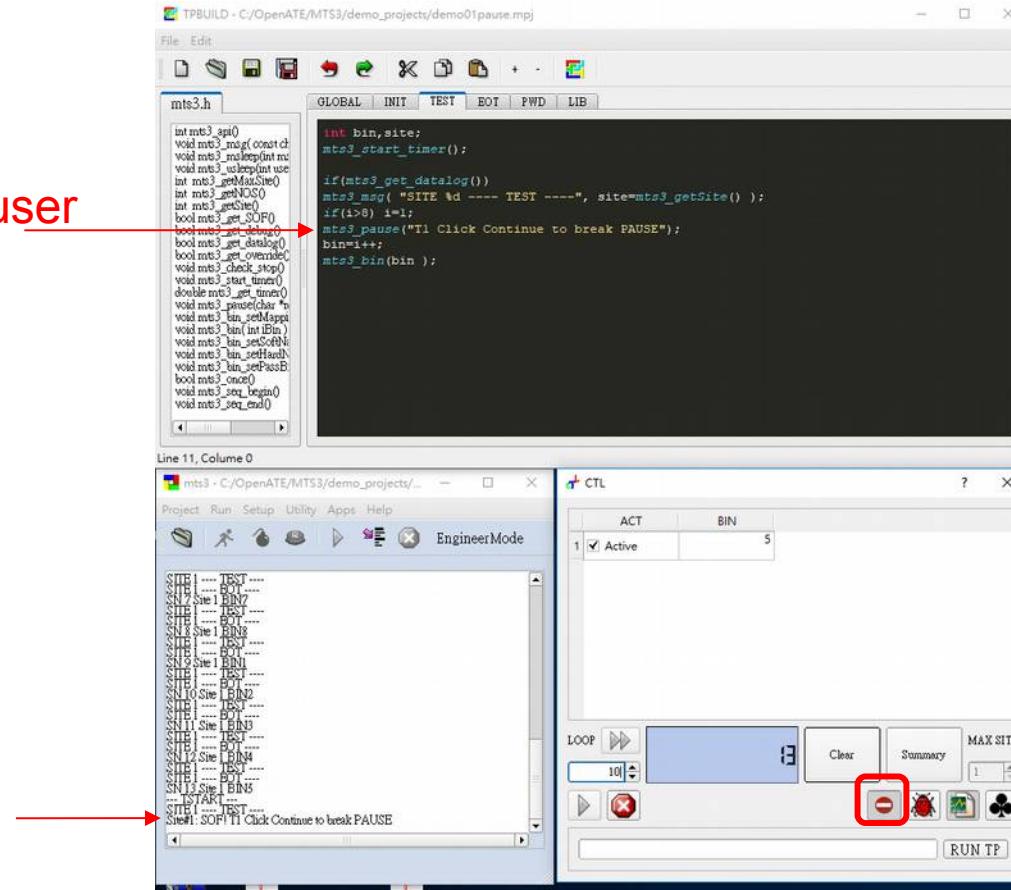
- You can press “Loop” button to run several times.



# Step6

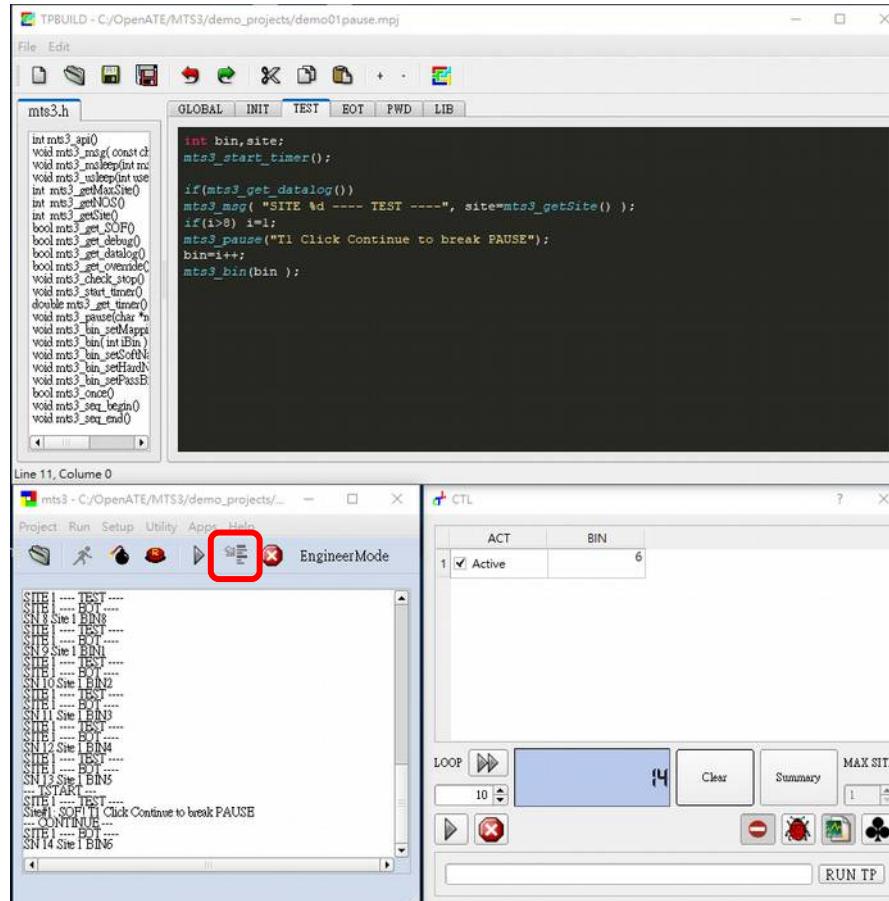
- Now, we will test mts3\_pause() function.
- You must press “Pause” button and press “Tstart” button to run the test program once.

Pause execution of test program until user  
hits continue button.



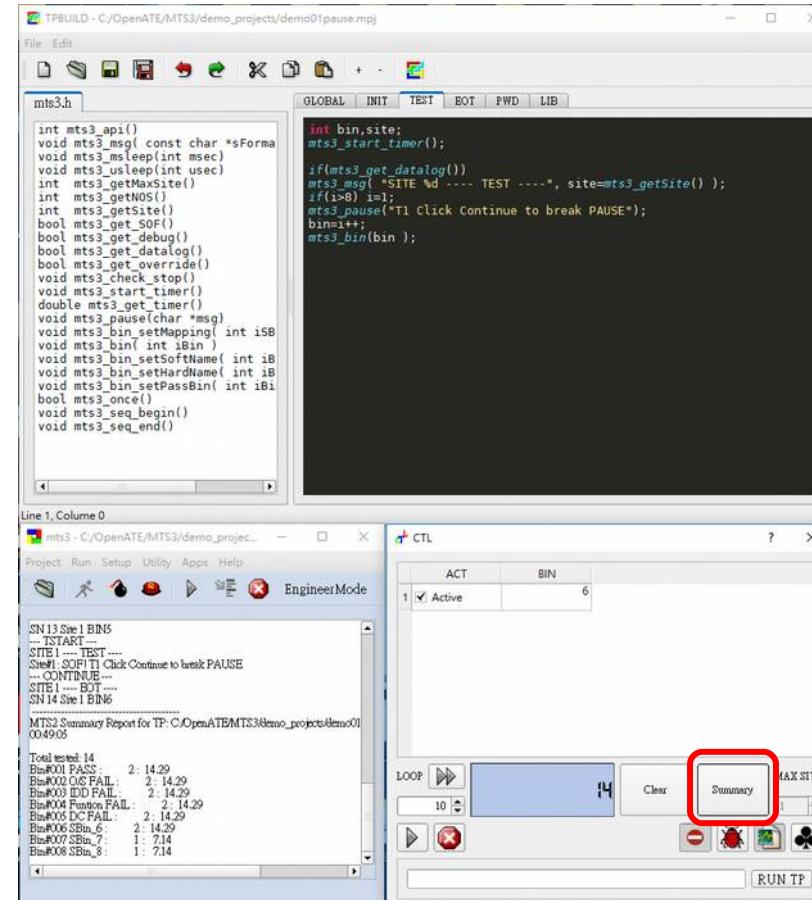
# Step7

- Press “Continue” button to continue execution.



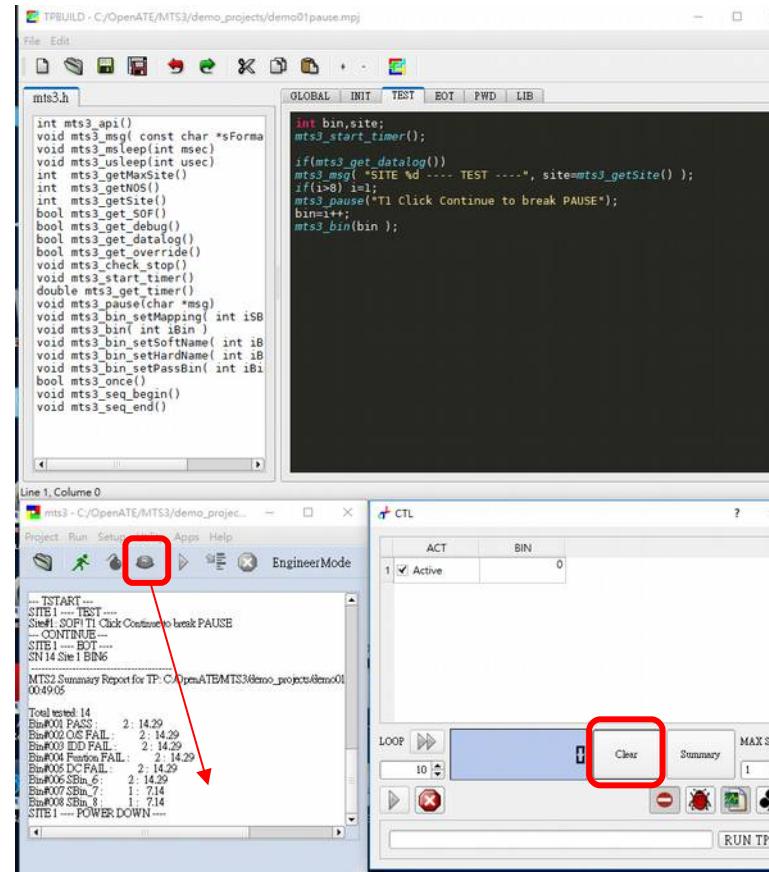
# Step8

- Press “Summary” button to see summary information.



# Step9

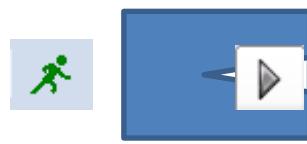
- Press “Reset” button to finish execution.
- Press “Clear” button to clear the number of times.



# Demo Project2

Step-by-step

# Demo TIMER



GLOBAL INIT TEST EOT PWD LIB

```
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

ACT BIN

1	<input checked="" type="checkbox"/> Active	1
2	<input checked="" type="checkbox"/> Active	2

LOOP  2 Clear Summary MAX SITE 2

RUN TP

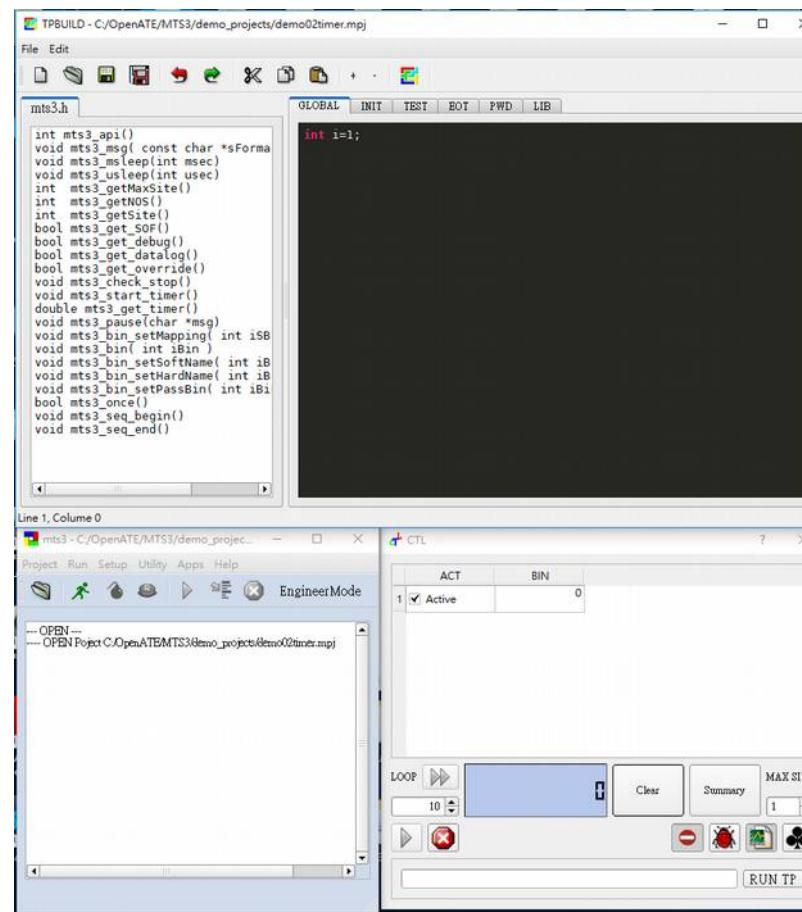
Loading - PE16UserManu.pdf  
Loading - PE32UserManu.pdf  
Loading - QSPIUserManu.pdf  
Loading - SMU64UserManu.pdf  
Run TSP:  
--- OPEN ---  
--- OPEN Project C:/OpenATE/MTS3/demo\_projects/demo02timer.mpj  
## UT TPBUILD.dll exec....  
UTY C:/OpenATE/mts3/ut/TPBUILD.dll run.....  
--- OPEN Project C:/OpenATE/MTS3/demo\_projects/demo02timer.mpj  
DEMO Timer function  
DEMO Timer function  
SITE 2 ---- INIT ----  
SITE 1 ---- INIT ----  
--- TSTART ---  
SITE 2 ---- TEST ----  
SITE 1 ---- TEST ----  
TIMER 10.00 mS,SITE 1  
---- TIMER 10.000000 ----I AM SITE 1  
TIMER 20.00 mS,SITE 2  
---- TIMER 20.000000 ----I AM SITE 2  
SITE 1 ---- EOT ----  
SITE 2 ---- EOT ----  
SN 2 Site 2 BIN2  
SN 2 Site 1 BIN1

# 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

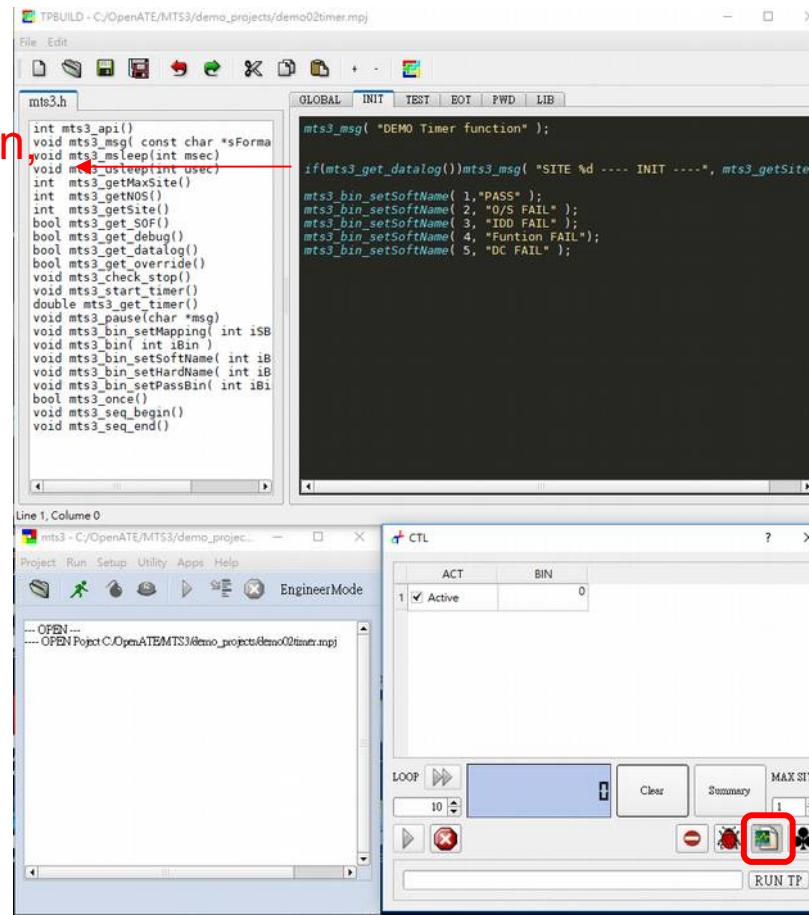
- Open MTS3
- Open project “C:\OpenATE\MTS3\demo\_projects\demo02timer.mpj”
- Open TPBUILD
  - Utility → TPBUILD



# Step2

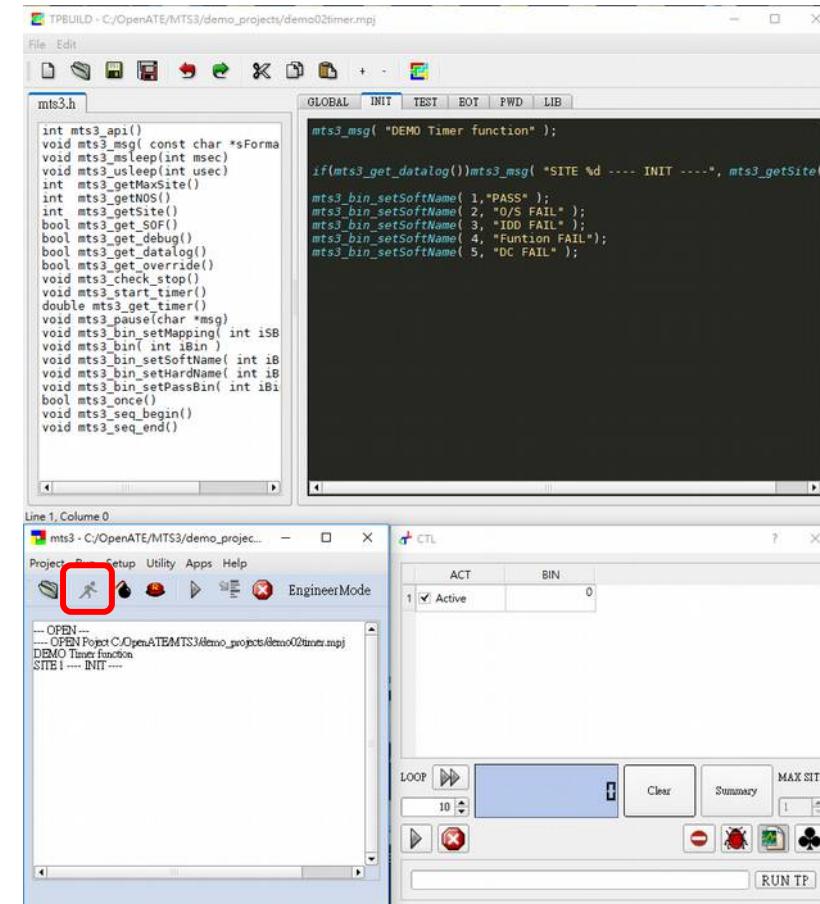
- You can press “Datalog” button to see some information.

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



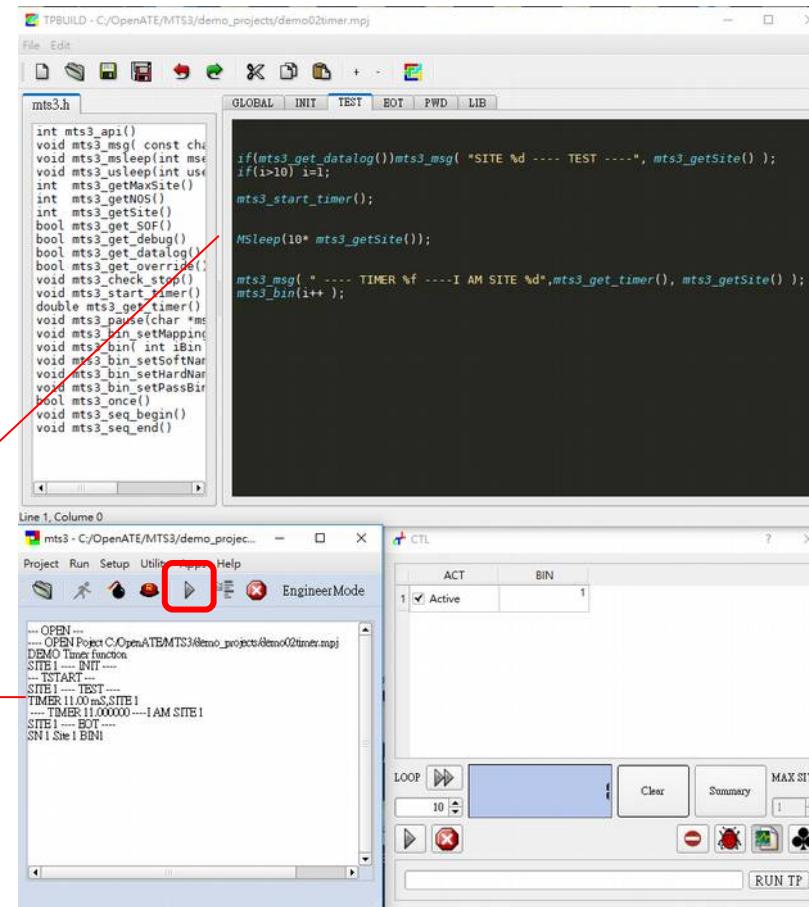
# Step3

- Press “Exec” button to initialize.



# Step4

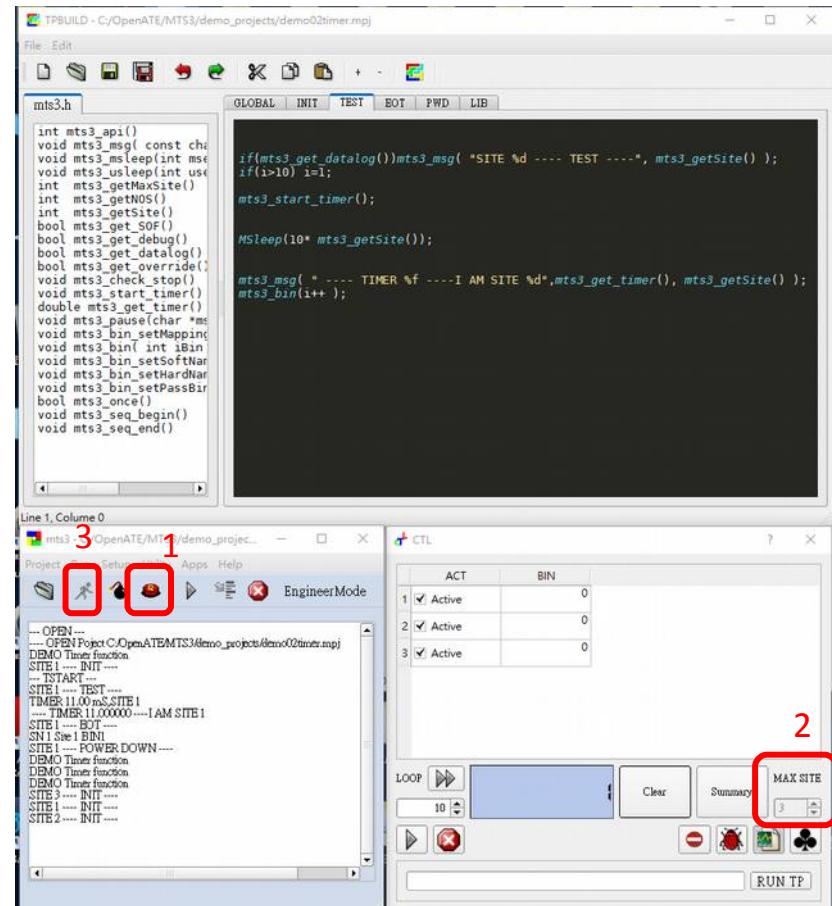
- Press “Tstart” button to run the test program once.



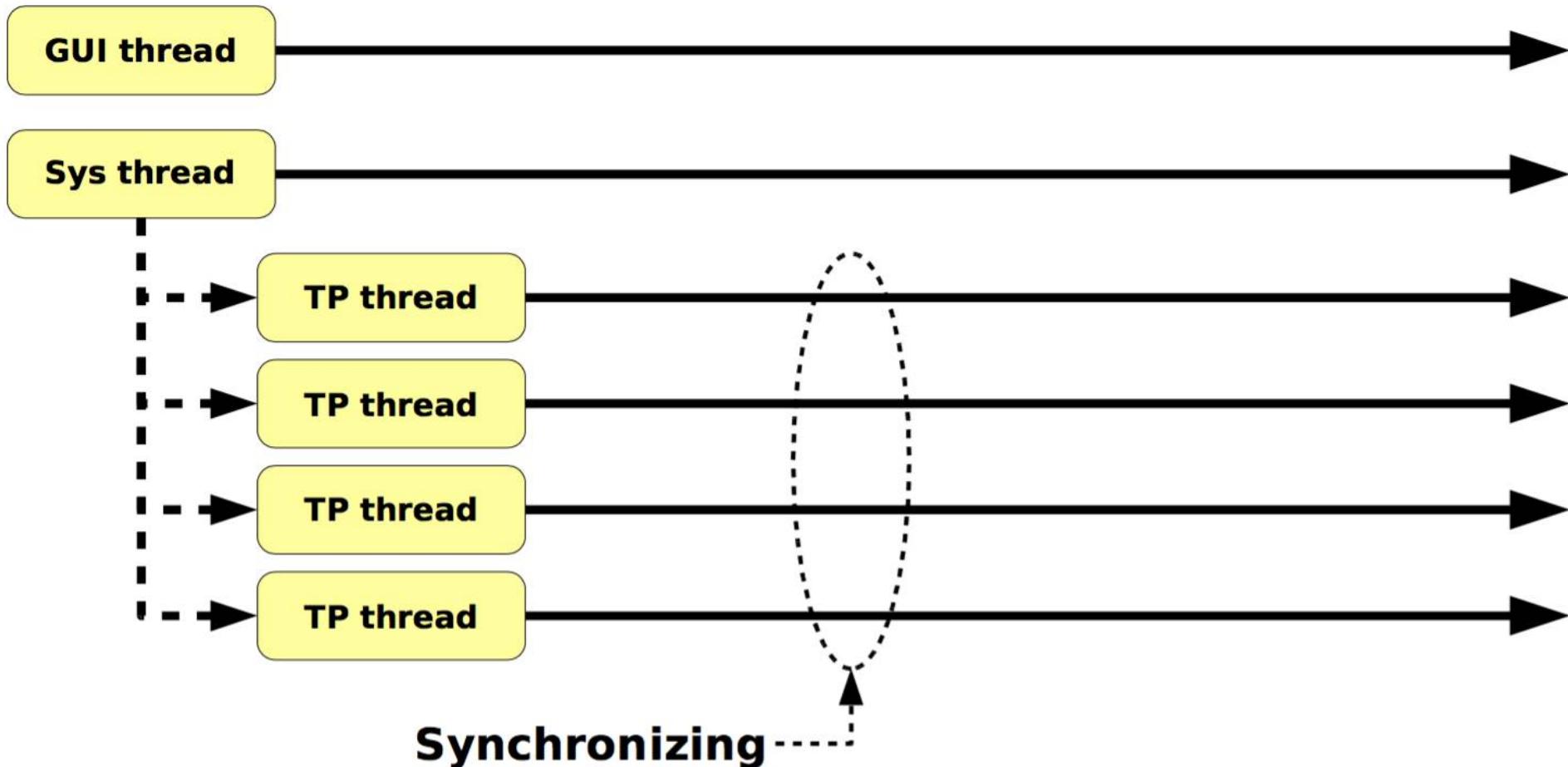
Site1 waits for  $10 * 1$  ms

# Step5

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

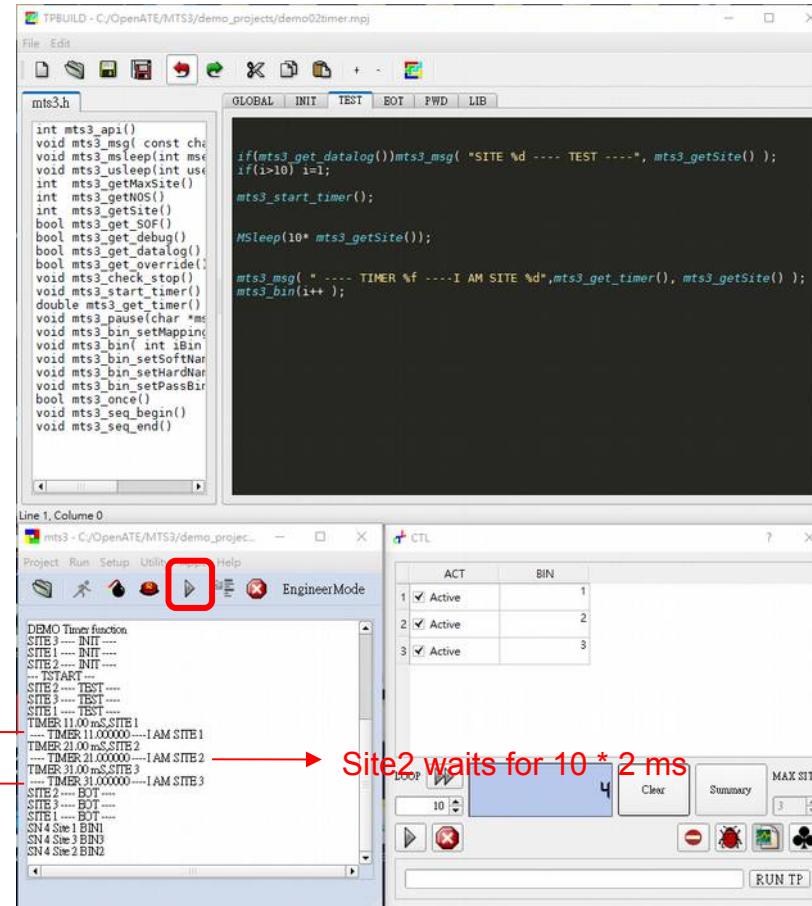


# Threading



# Step6

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



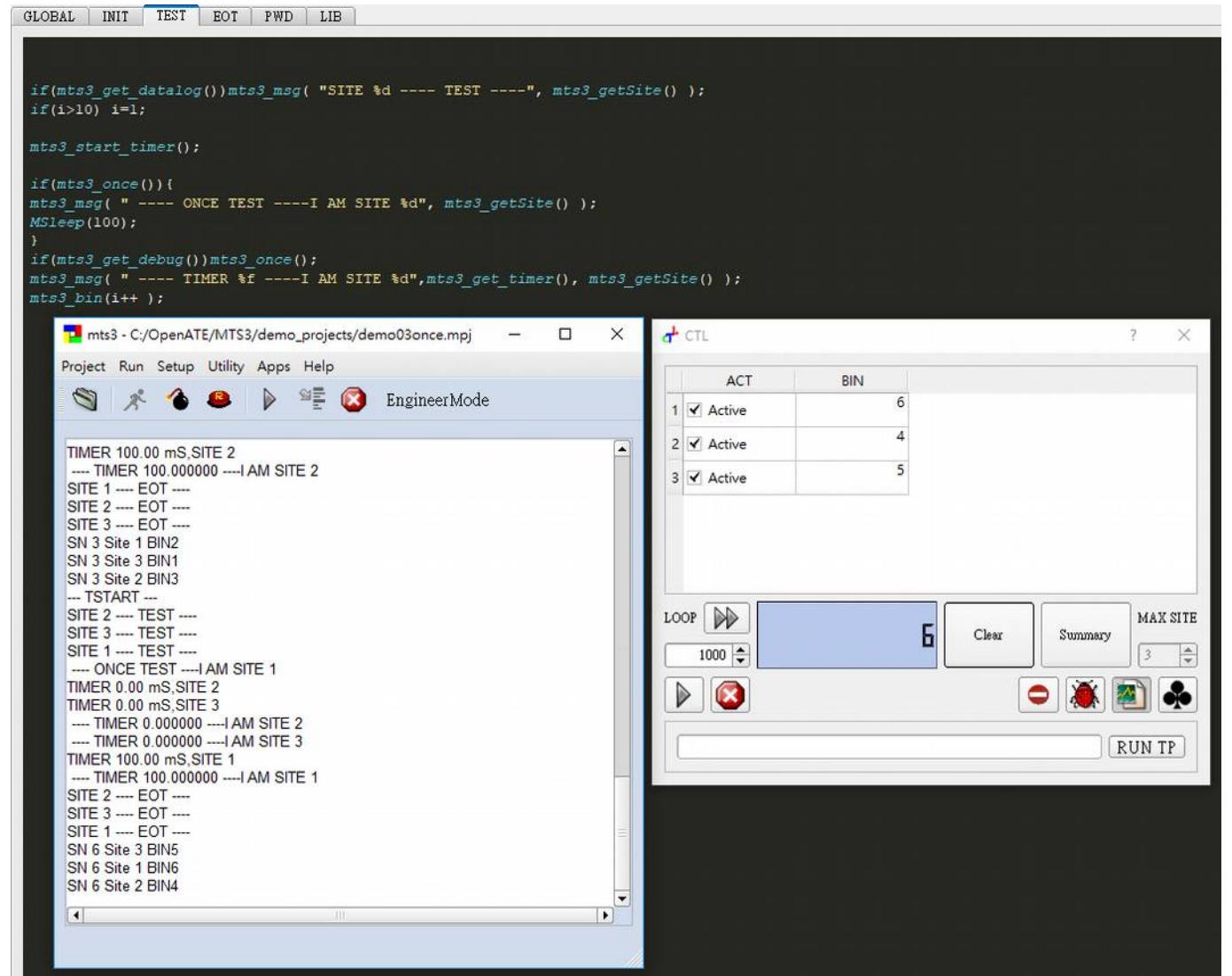
# Demo Project3

Step-by-step

# Demo Project\_3

## Demo

■



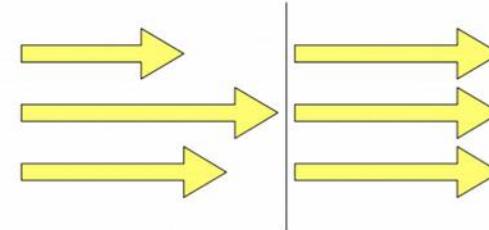
# Introduction to API

- **bool mts3\_once();**
  - Sometimes for a multi-site application, a piece of code may need to be executed only once by one thread.
  - Usage:

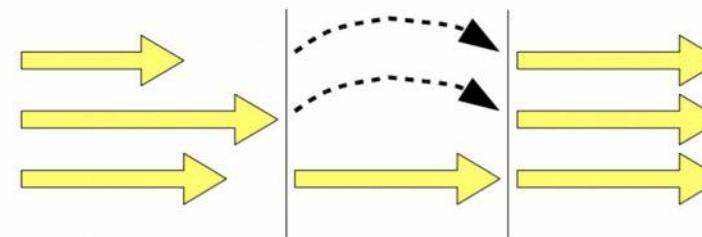
```
if ( mts3_once() ) {  
    // statements in this block will be executed only once by one thread.  
}
```
  - `mts3_once()` is also used to sync the execution among test threads.

# Synchronizing

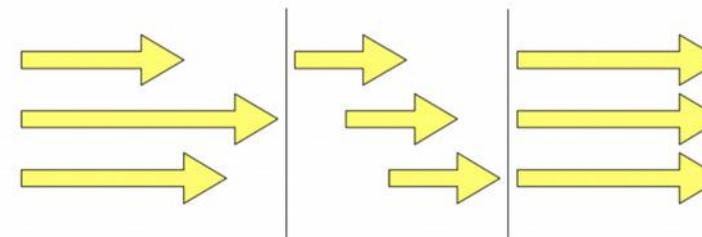
**Synch**



**Once**

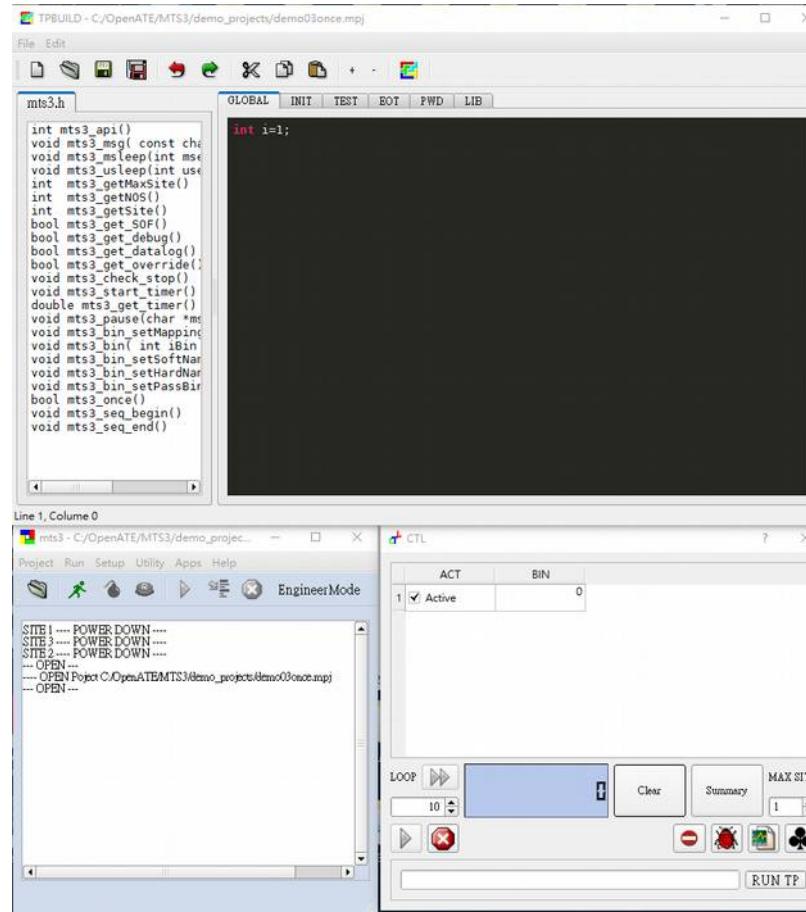


**Sequence**



# Step1

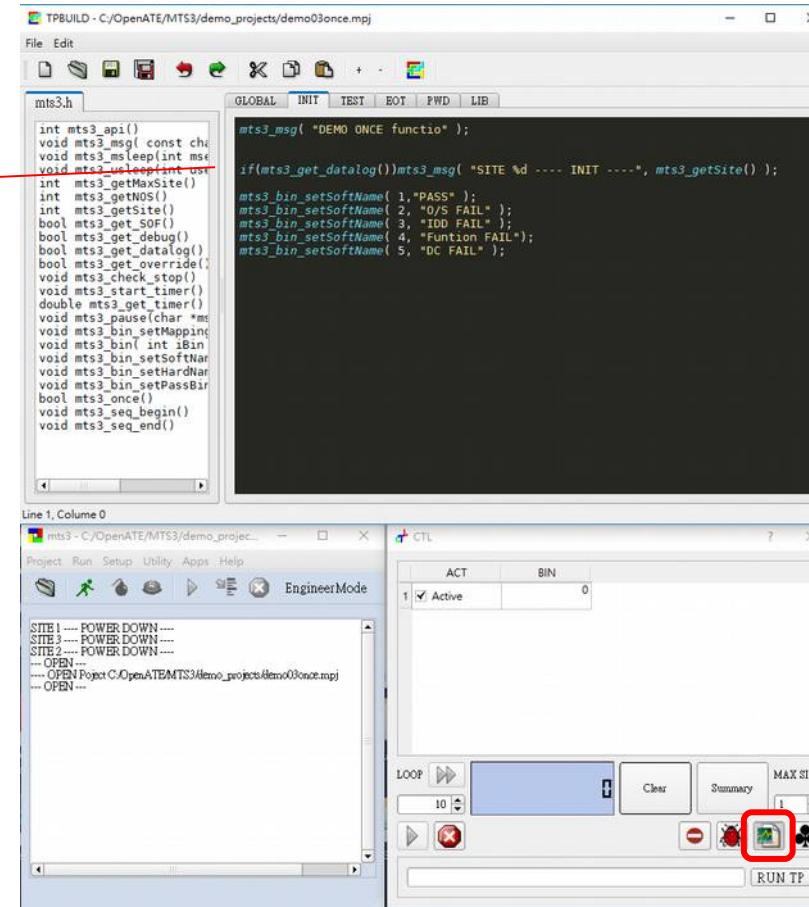
- Open MTS3
- Open project “C:\OpenATE\MTS3\demo\_projects\demo03once.mpj”
- Open TPBUILD
  - Utility → TPBUILD



# Step2

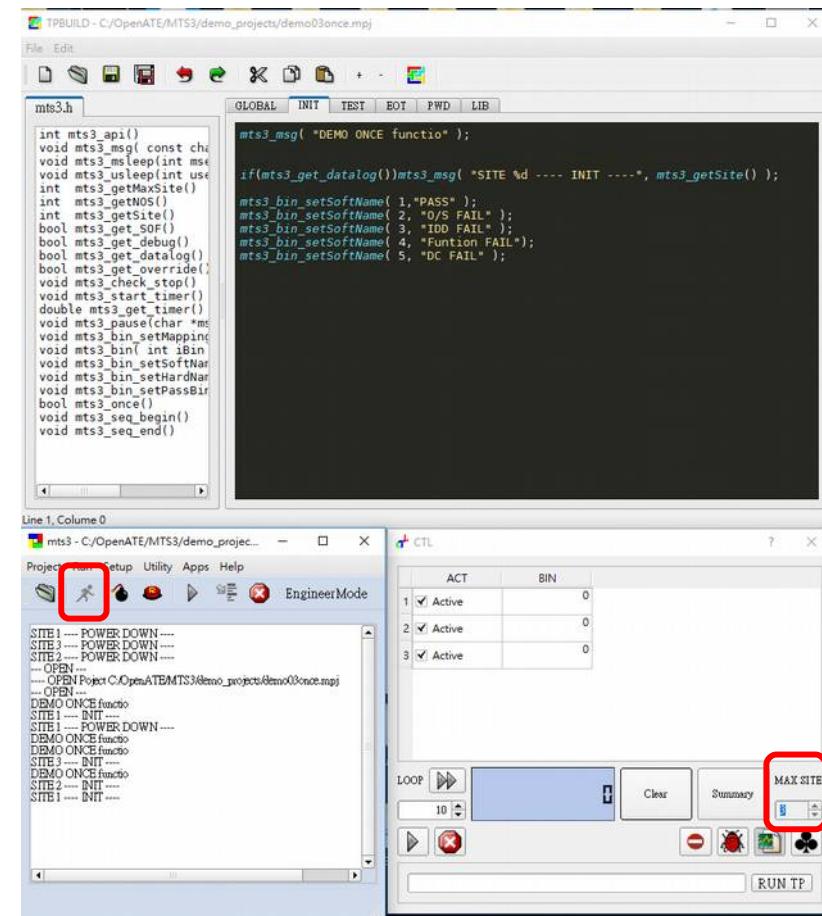
- You can press “Datalog” button to see some information.

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



# Step3

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



# Step4

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

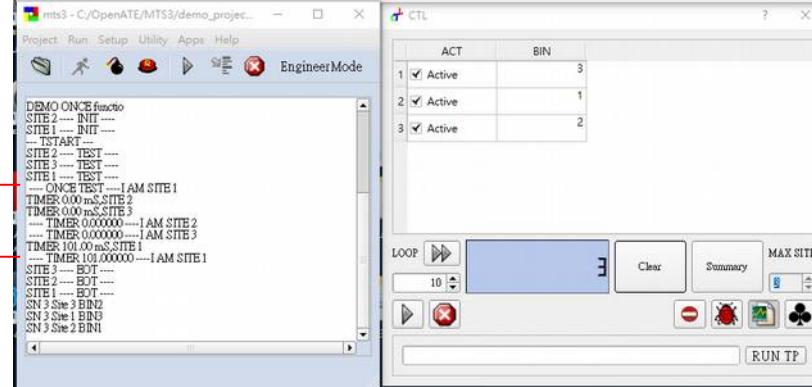
Statements in this block will be executed only once by one thread.

The screenshot shows a software interface with two main windows. The top window is titled 'TPBUILD - C:/OpenATE/MTS3/demo\_projects/demo03once.mpj' and displays the contents of the file 'mts3.h'. The code includes several functions such as mts3\_api(), mts3\_msleep(), mts3\_getMaxSize(), mts3\_getSite(), mts3\_getSOF(), mts3\_get\_debug(), mts3\_get\_datalog(), mts3\_start\_stop(), mts3\_start\_timer(), mts3\_stop(), mts3\_pause(), mts3\_bin\_setMapping(), mts3\_bin\_setSoftbar(), mts3\_bin\_setHardbar(), mts3\_bin\_setPassBar(), mts3\_once(), mts3\_seq\_begin(), and mts3\_seq\_end(). A red arrow points from the text 'Statements in this block will be executed only once by one thread.' to the line 'if(mts3\_once()) {'. The bottom window is titled 'mts3 - C:/OpenATE/MTS3/demo\_project...' and shows a script with the following content:

```
DEMO ONCE function
SITE2 --- INIT ---
SITE1 --- INIT ---
TSTART
SITE1 --- TEST ---
SITE2 --- TEST ---
SITE3 --- TEST ---
SITE1 --- TEST ---
--- ONCE TEST --- I AM SITE1
TIMER 0.00 ms SITE2
TIMER 0.00 ms SITE3
--- TIMER 0.000000 --- I AM SITE2
--- TIMER 0.000000 --- I AM SITE3
TIMER 10.00 ms SITE1
--- TIMER 10.000000 --- I AM SITE1
SITE3 --- BOT
SITE2 --- BOT
SN3 Site3 BIN2
SN3 Site1 BIN3
SN3 Site2 BIN1
```

It is executed only once by one thread.

So only site1 waits for 100ms.



# Demo Project4

Step-by-step

# Demo Project\_4

## Demo

The screenshot shows the OpenATE MTS3 software interface. The top menu bar includes GLOBAL, INIT, TEST (selected), EOT, PWD, and LIB. The main window displays C code for a project named 'demo04seq.mpj'. The code uses the mts3 library to perform a sequence of tests across three sites. A red box highlights the section of the log output where the SEQ function is used to execute multiple tests sequentially. Another red box highlights the section where the SEQ function is removed, showing the original looped test sequence.

**Before using SEQ function:**

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

//if(mts3_once())
//mts3_msg( " NO SEQ ----I AM SITE %d", mts3_getSite() );
//MSleep(1);
//mts3_msg( " ---- SYN TEST 1 ----I AM SITE %d", mts3_getSite() );
//MSleep(1);
//mts3_msg( " ---- SYN TEST 2 ----I AM SITE %d", mts3_getSite() );
//MSleep(1);
//mts3_msg( " ---- SYN TEST 3 ----I AM SITE %d", mts3_getSite() );
//MSleep(1);
//mts3_msg( " ---- SYN TEST 4 ----I AM SITE %d", mts3_getSite() );
//MSleep(1);
//mts3_msg( " ---- SYN TEST 5 ----I AM SITE %d", mts3_getSite() );
//MSleep(1);

if(mts3_once())
mts3_msg( " WITH SEQ ----I AM SITE %d", mts3_getSite() );
mts3_seq_begin();

mts3_msg( " ---- SYN TEST 6 ----I AM SITE %d", mts3_getSite() );
MSleep(1);
mts3_msg( " ---- SYN TEST 7 ----I AM SITE %d", mts3_getSite() );
MSleep(1);
mts3_msg( " ---- SYN TEST 8 ----I AM SITE %d", mts3_getSite() );
mts3_seq_end();
```

**After using SEQ function:**

```
-----  
SITE 2 ---- INIT ----  
DEMO SEQ function  
DEMO SEQ function  
SITE 1 ---- INIT ----  
SITE 3 ---- INIT ----  
--- TSTART ---  
SITE 1 ---- TEST ----  
--- SYN TEST 1 ----I AM SITE 1  
SITE 2 ---- TEST ----  
--- SYN TEST 1 ----I AM SITE 2  
SITE 3 ---- TEST ----  
--- SYN TEST 1 ----I AM SITE 3  
--- SYN TEST 2 ----I AM SITE 1  
--- SYN TEST 2 ----I AM SITE 2  
--- SYN TEST 2 ----I AM SITE 3  
--- SYN TEST 3 ----I AM SITE 1  
--- SYN TEST 3 ----I AM SITE 2  
--- SYN TEST 3 ----I AM SITE 3  
--- SYN TEST 4 ----I AM SITE 1  
--- SYN TEST 4 ----I AM SITE 2  
--- SYN TEST 4 ----I AM SITE 3  
--- SYN TEST 5 ----I AM SITE 1  
--- SYN TEST 5 ----I AM SITE 2  
--- SYN TEST 5 ----I AM SITE 3  
WITH SEQ ....I AM SITE 1  
--- SYN TEST 6 ----I AM SITE 3  
--- SYN TEST 7 ----I AM SITE 3  
--- SYN TEST 8 ----I AM SITE 3  
--- SYN TEST 6 ----I AM SITE 1  
--- SYN TEST 7 ----I AM SITE 1  
--- SYN TEST 8 ----I AM SITE 1  
--- SYN TEST 6 ----I AM SITE 2  
--- SYN TEST 7 ----I AM SITE 2  
--- SYN TEST 8 ----I AM SITE 2  
--- SYN TEST 6 ----I AM SITE 3  
--- SYN TEST 7 ----I AM SITE 3  
--- SYN TEST 8 ----I AM SITE 3  
SITE 3 ---- EOT ----  
SITE 1 ---- EOT ----  
SITE 2 ---- EOT ----  
SN 3 Site 2 BIN3  
SN 3 Site 3 BIN1  
SN 3 Site 1 BIN2
```

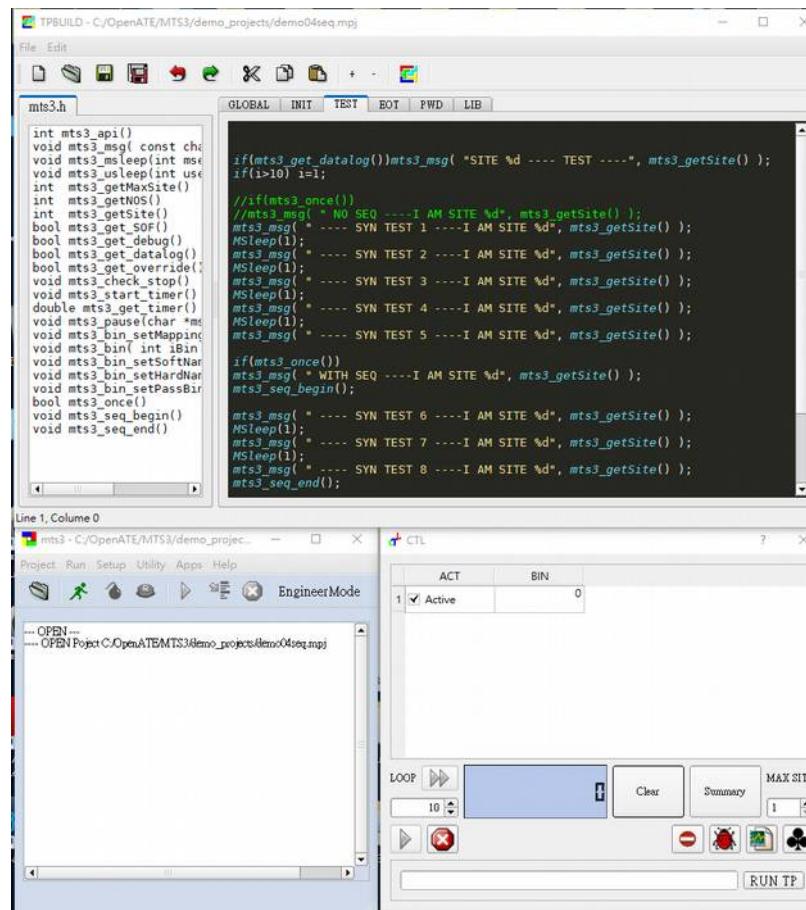
# Introduction to API

- `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
```

# Step1

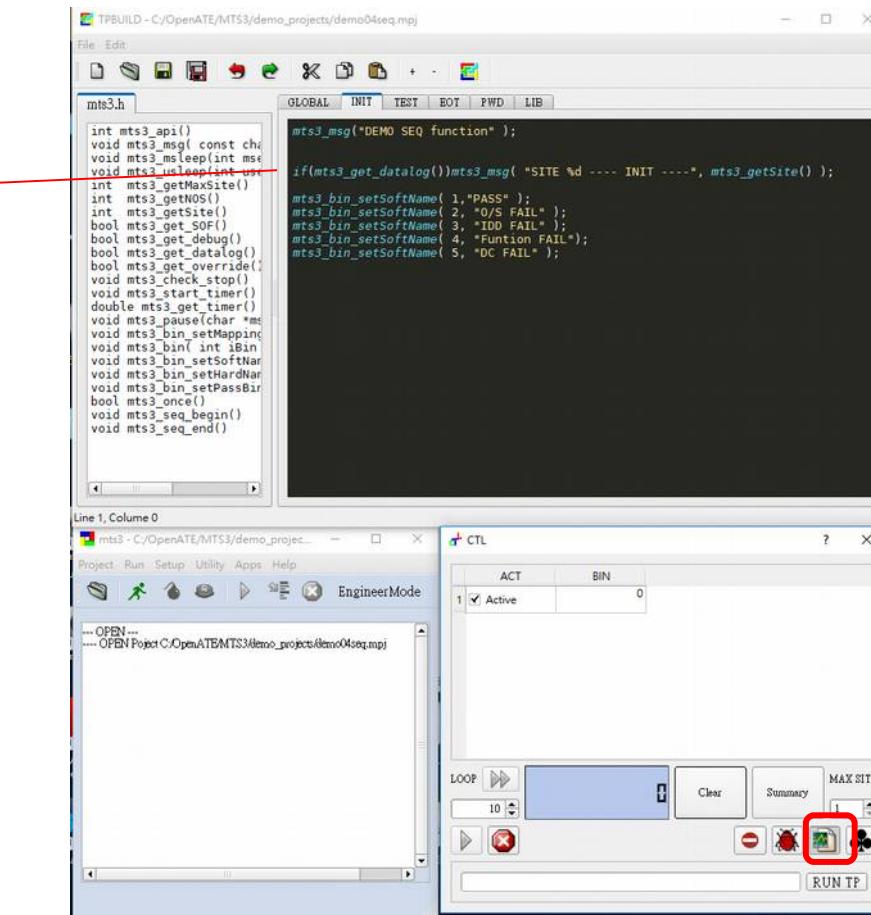
- Open MTS3
- Open project “C:\OpenATE\MTS3\demo\_projects\demo04seq.mpj”
- Open TPBUILD
  - Utility → TPBUILD



# Step2

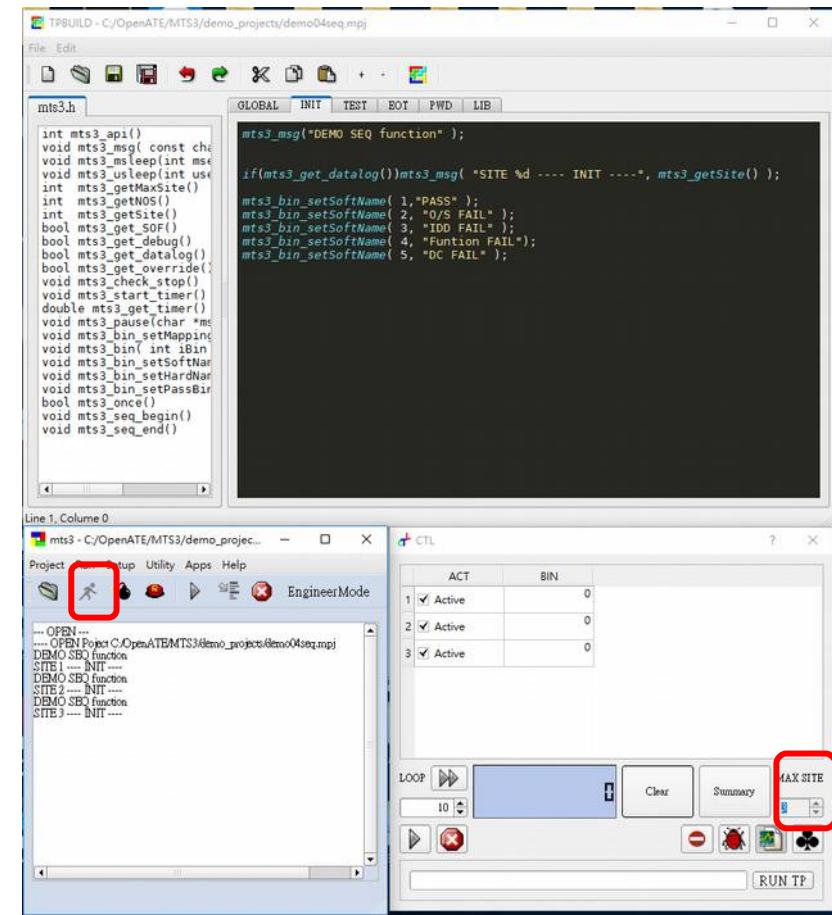
- You can press “Datalog” button to see some information.

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



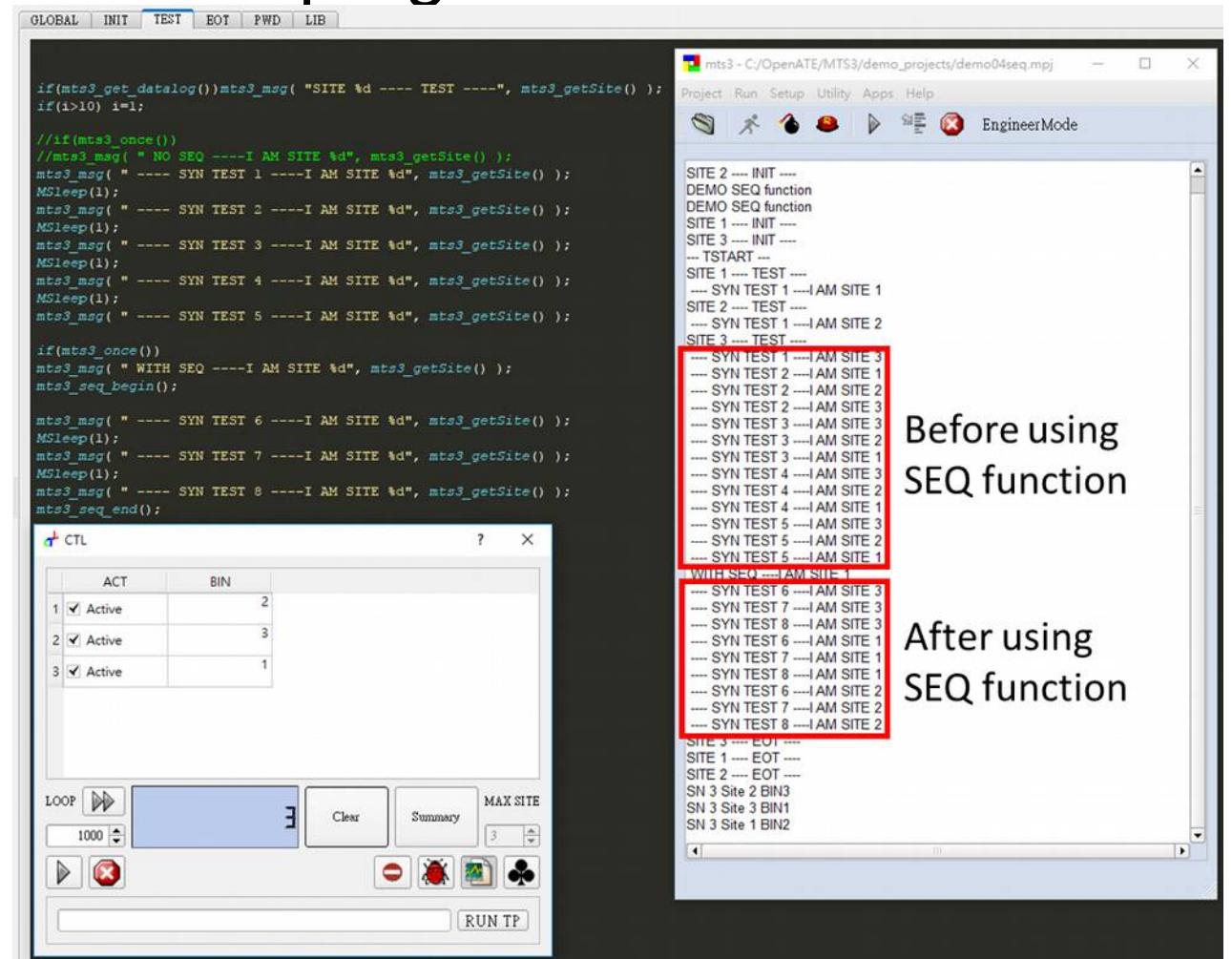
# Step3

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



# Step4

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



# Demo Project5

Step-by-step

# Demo Project\_5

## 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 showing C-like pseudocode:

```
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++ );
```

Below the code editor is a terminal window titled "mts3 - C:/OpenATE/MTS3/demo\_projects/demo05chkstop...". It displays the following text:

```
... TSTART ...
SITE 1 ---- TEST ----
SITE 1 push STOP TO BREAK infinite loop ----
... STOP ...
SITE 1 ---- EOT ...
```

To the right of the terminal is a control panel window titled "CTL". It contains a table with one row:

ACT	BIN
1 <input checked="" type="checkbox"/> Active	0

Below the table are several control buttons and settings:

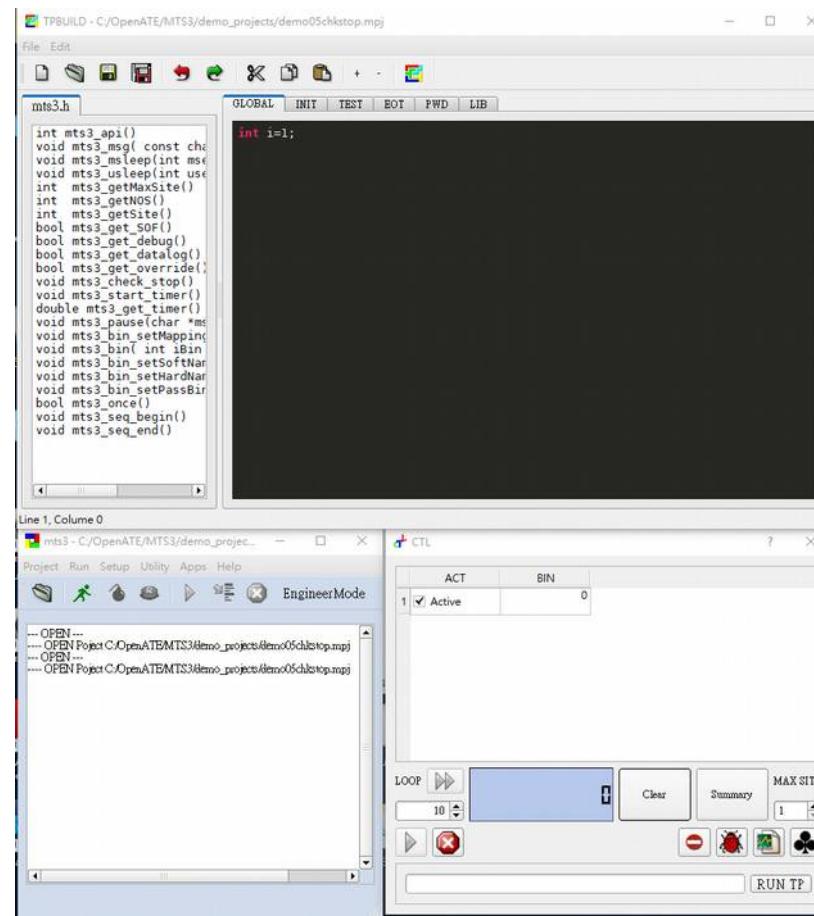
- LOOP: A button with a double arrow and a dropdown set to 1000.
- Clear: A button with a red X.
- Summary: A button with a document icon.
- MAX SITE: A dropdown set to 1.
- Other buttons include a red minus sign, a red circle, a green checkmark, and a black club symbol.
- A "RUN TP" button is located at the bottom right.

# 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 infinit 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 OpenATE MT53 software interface. The top window is a code editor titled "mts3.h" with C code. A red arrow points from the text "infinite loop" to the line of code "while(1){". The bottom window is a terminal titled "Line 1, Column 0" showing command history and a log message: "SITE1 push STOP TO BREAK infinite loop ---". To the right are two control panels: "CTL" showing a table with one row and "LOOP" with various buttons and settings.

```
TPBUILD - C:/OpenATE/MTS3/demo_projects/demo05chstop.mpj
File Edit
mts3.h
GLOBAL INIT TEST BOT PWD LIB
int mts3_api()
void mts3_msg( const char *msg );
void mts3_msleep(int ms);
void mts3_usleep(int us);
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_stop();
void mts3_start();
void mts3_start_timer();
double mts3_get_timer();
void mts3_pause(char *msg);
void mts3_bin_setMapping();
void mts3_binf_int iBin;
void mts3_bin_setSoftNar();
void mts3_bin_setHardNar();
void mts3_bin_setPassBir();
bool mts3_once();
void mts3_seq_begin();
void mts3_seq_end()

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++);

Line 1, Column 0
mts3 - C:/OpenATE/MTS3/demo_projects...
Project Run Setup Utility Apps Help
EngineerMode
OPEN -->
OPEN Project C:/OpenATE/MTS3/demo_projects/demo05chstop.mpj
OPEN Project C:/OpenATE/MTS3/demo_projects/demo05chstop.mpj
DEMO CHECKSTOP
SITE1 ---- INIT ----
--- TSTART ---
SITE1 ---- TEST ---
SITE1 push STOP TO BREAK infinite loop ---
```

# Step3

- Push STOP to break infinite loop

The screenshot shows the OpenATE software interface. At the top is a code editor window titled "mts3.h" with the following C code:

```
int mts3_api();
void mts3_msleep(const char *msg);
void mts3_msleep(int msec);
void mts3_msleep(int usec);
int mts3_getMaxSite();
int mts3_getIOS();
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();
void mts3_binf( int iBin );
void mts3_bin_setSoftNar();
void mts3_bin_setHardNar();
void mts3_bin_setPassBin();
bool mts3_once();
void mts3_seq_begin();
void mts3_seq_end();
```

Below the code editor is a test console window titled "mts3 - C:/OpenATE/MTS3/demo\_projects/demo05chkstop.mjt". It displays the following log output:

```
... OPEN ...
... OPEN Project C:/OpenATE/MTS3/demo_projects/demo05chkstop.mjt
... OPEN ...
... OPEN Project C:/OpenATE/MTS3/demo_projects/demo05chkstop.mjt
DMO CHKSTOP
SITE1---- INT1----
... START ...
SITE1---- TEST ...
SITE1---- push STOP TO BREAK infinite loop ----
SITE1---- END ...

```

At the bottom right of the test console window, there is a red box highlighting the "EngineerMode" button.

# Demo Project6

# Demo Project\_6

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

The screenshot shows the OpenATE MTS3 software interface. At the top, there is a menu bar with tabs: GLOBAL, INIT, TEST (which is selected), EOT, PWD, and LIB. Below the menu is a code editor window displaying C code related to site testing. To the right of the code editor is a log output window showing test results for three sites. At the bottom right is a control panel window titled 'CTL' containing a table of active sites and various control buttons.

Code in the editor:

```
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++ );
```

Log output window:

```
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
```

CTL Control Panel:

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

Buttons and controls include: LOOP (1000), Clear, Summary, MAX SITE (3), RUN TP, and several status indicators (red, green, yellow).

# Demo Project\_6

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

The screenshot shows the OpenATE MTS3 software interface. The top part is a code editor with tabs: GLOBAL, INIT, TEST (selected), EOT, PWD, LIB. The code in the TEST tab is:

```
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++ );
```

The bottom part is the CTL (Control) window. It has two panes: a log pane on the left and a control pane on the right.

**Log Pane:**

```
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
```

**CTL Pane:**

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

Buttons and controls in the CTL pane include: LOOP (1000), Clear, Summary, MAX SITE (set to 3), and RUN TP.

# Introduction to API

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

# Reference

# Reference

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