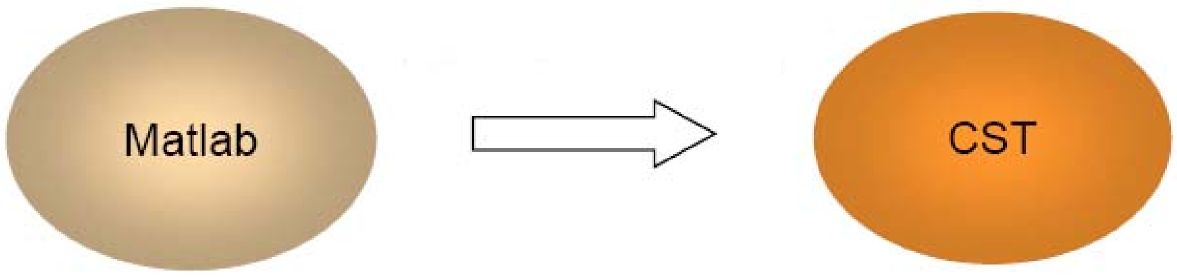
Application Note – Calling CST Studio from Matlab

Abstract

This application note gives background information on linking CST Studio to Matlab, by calling CST directly from Matlab.



For information on how to call Matlab from within CST Studio, please see the separate Application Note CST_USER_NOTE__MATLAB_COM_DDE.pdf .

Methods

There are two main ways to call CST Studio from within Matlab:

1) launch CST from Matlab command line and use a separate VBA file to instruct CST what operations to do
2) run a Matlab .m file containing CST Studio commands

1. Launch CST from Matlab command line

This is the simplest way to run CST from within Matlab. It is appropriate when the CST model name and operations to be performed are known in advance and do not need to change.

At the Matlab command line, enter:

```
>> ! "CST_DS_Path\CST DESIGN ENVIRONMENT.exe" –m cmdfile.bas
```

where *`CST_DS_Path`* is the path to the CST STUDIO SUITE installation directory, and *`cmdfile`*`.bas` is a Basic (VBA) file containing the commands to be executed within CST STUDIO SUITE.

1.1. Example 1: launch CST, open an existing model, solve, save

For example, when *`CST_DS_Path`* is

`ProgramFiles\CSTSTUDIO SUITE 2006`

and the command file name (including path) is

`C:\Work\Start_CST.bas`

the Matlab command line would look like follows:

```
>> ! "C:\ProgramFiles\CSTSTUDIO SUITE 2006\CST DESIGN
ENVIRONMENT.exe" –m C:\Work\Start_CST.bas
```

The contents of the `Start_CST.bas` file could look as follows:

```
Sub Main
    OpenFile("C:\work\test1.cst”)
    Solver.Start
    Save
End Sub
```

This simple VBA script opens the CST model called test1.cst, starts the solver, saves the model and then exits.

2. Launch and control CST from a Matlab .m file

In this case, all the commands that need to be executed in CST, instead of being contained in a VBA .bas file are contained in a Matlab .m file. They could as well be entered directly from the Matlab prompt.
This technique has the advantage that it offers complete control of CST MWS from within Matlab. Various operations, like solving several models, performing parameter studies, changing a model, saving or discarding MWS results, etc. are therefore accessible.

The user has access to all CST-specific VBA commands, by means of the Matlab command `invoke`.

2.1. Starting CST DESIGN ENVIRONMENT and opening an MWS window

The first step is typically **starting the CST DESIGN ENVIRONMENT**. To perform this operation, the following command syntax is used:

```
cst = actxserver('CSTStudio.application')
```

“`cst`” can be replaced by any other variable name.

To **open a new MWS window** in CST DS, the following syntax is used:

```
mws = invoke(cst, 'NewMWS')
```

where “`mws`” can be replaced by any other variable name.

To **open an existing MWS model**, you can (just as if you were working interactively within CST DS):
- open the model directly into CST DS:

```
mws = invoke(cst, 'OpenFile', ‘filename.cst');
```

- or, first open an empty MWS window, then open the model in this window:

```
mws = invoke(cst, 'NewMWS')
invoke(mws, 'OpenFile', ‘filename.cst');
```

2.2. Translating CST-specific VBA commands into Matlab

Any set of CST-specific VBA commands can be issued directly from Matlab. For example, the VBA code for the generation of a brick:

```
With Brick
        .Reset
        .Name “Brick1”
        .Component “component1”
        .Xrange “0”, “1”
        .Yrange “0”, “1”
        .Zrange “0”, “1”
        .Material “Vacuum”
        .Create
End With
```

translates into the following Matlab code:

```
brick = invoke(mws, 'Brick');
     % create a brick in MWS
     invoke(brick, 'Reset');
     invoke(brick, 'Name', 'Brick1');
     invoke(brick, 'Component', 'component1');
     invoke(brick, 'Xrange', '0', '1');
     invoke(brick, 'Yrange', '0', '1');
     invoke(brick, 'Zrange', '0', '1');
     invoke(brick, 'Material', 'Vacuum');
     invoke(brick, 'Create');

     % ... possibly create other bricks here
release(brick);
```

Note: if several bricks need to be created, the pair of commands `brick = invoke(mws, 'Brick')` and `release(brick)` only need to be issued once.

2.3. Useful commands – some examples

Find out the application name and version

```
     app = invoke(mws, 'GetApplicationName')
     ver = invoke(mws, 'GetApplicationVersion')
```

Set the units

```
     units = invoke(mws,'Units');
     invoke(units, 'Geometry','mm');
     invoke(units, 'Frequency','ghz');
     invoke(units, 'Time','s');
```

Set the value of a variable

```
     len1=40.00;
     invoke(mws, 'StoreParameter','len1',len1);
          % "mws" is the Matlab name of the MWS object, see &2.1
```

Solver commands examples

```
     solver = invoke(mws, 'Solver');
     invoke(solver, 'FrequencyRange', fr1, fr2);          % Set the
     frequency range
     invoke(solver, 'Start');
```

Working with 1D results

```
     result = invoke(mws, 'Result1D', 'a1(1)1(1)');  % Example for
                                                      % S11 linear
     numOfValue = invoke(result, 'GetN');  % Might be used to
                                           % initialize variables

     invoke(result, 'Save', 'C:\tmp\filename');
     A = importdata('C:\tmp\filename', ' ', 4)
     x = A.data(:, 1);     % x-data column
     y = A.data(:, 2);     % y-data column
```

2.4. Example 1 again: launch CST, open an existing model, solve, save

To reach the same effect as in the example shown in paragraph 1.1, the following set of commands need to be entered into the .m file:

```
      cst = actxserver('CSTStudio.application')

      mws = invoke(cst, 'NewMWS')
      invoke(mws, 'OpenFile', 'C:\work\test1.cst');
      solver = invoke(mws, 'Solver');
      invoke(solver, 'Start');
      invoke(mws, 'Save');

      release(solver);
      release(mws);
      release(cst);
```

2.5. Example 2: Open an MWS file, perform a parameter study, store a result in a Matlab vector

The following code can be used to perform this operations. (The code is contained in the attached file **studio3.m**. It refers to the CST model **tsplitter.cst**.)

```
%matlab-example: tested 09-Jan 2008 with with MATLAB 7.5 (R2007b)
% Open an MWS file, perform a parameter study,
%    store a result in a Matlab vector

% To test, first copy the file "tsplitter.cst"
%    in the directory "C:\CST_Test"

% Start CST Studio
cst = actxserver('CSTStudio.application')
mws = invoke(cst, 'NewMWS')
app = invoke(mws, 'GetApplicationName')
ver = invoke(mws, 'GetApplicationVersion')

% Open the MWS model
invoke(mws, 'OpenFile', 'C:\CST_Test\tsplitter.cst');

for iii = 1:1:2       % Parameter sweep
      % Set value of parameter
      invoke(mws, 'StoreParameter','offset', iii);

      invoke(mws, 'Rebuild');

      % Start solver
      solver = invoke(mws, 'Solver');
      invoke(solver, 'Start');

      % Take over the result; here, as an example,
      %    the amplitude (linear) of S11
      result = invoke(mws, 'Result1D', 'a1(1)1(1)');
```

```
        numOfValue = invoke(result, 'GetN');  % Might be used to
                                              % initialize variables

        invoke(result, 'Save', 'C:\tmp\filename');
        A = importdata('C:\tmp\filename', ' ', 4)
        x(:, iii) = A.data(:, 1);       % x-data column
        y(:, iii) = A.data(:, 2);       % y-data column
    end

    invoke(mws, 'Save');
    invoke(mws, 'Quit');

    release(result);
    release(solver);
    release(mws);
    release(cst);
```

射 频 和 天 线 设 计 培 训 课 程 推 荐

易迪拓培训(www.edatop.com)由数名来自于研发第一线的资深工程师发起成立，致力并专注于微波、射频、天线设计研发人才的培养；我们于 2006 年整合合并微波 EDA 网(www.mweda.com)，现已发展成为国内最大的微波射频和天线设计人才培养基地，成功推出多套微波射频以及天线设计经典培训课程和 ADS、HFSS 等专业软件使用培训课程，广受客户好评；并先后与人民邮电出版社、电子工业出版社合作出版了多本专业图书，帮助数万名工程师提升了专业技术能力。客户遍布中兴通讯、研通高频、埃威航电、国人通信等多家国内知名公司，以及台湾工业技术研究院、永业科技、全一电子等多家台湾地区企业。

易迪拓培训课程列表：http://www.edatop.com/peixun/rfe/129.html

射频工程师养成培训课程套装

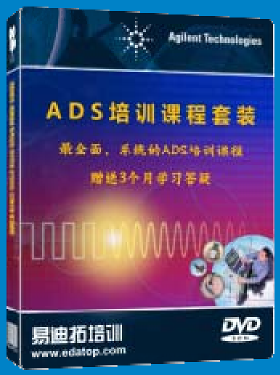
该套装精选了射频专业基础培训课程、射频仿真设计培训课程和射频电路测量培训课程三个类别共 30 门视频培训课程和 3 本图书教材；旨在引领学员全面学习一个射频工程师需要熟悉、理解和掌握的专业知识和研发设计能力。通过套装的学习，能够让学员完全达到和胜任一个合格的射频工程师的要求…

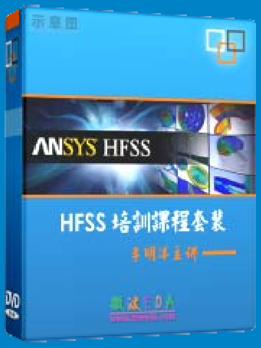
课程网址：http://www.edatop.com/peixun/rfe/110.html

ADS 学习培训课程套装

该套装是迄今国内最全面、最权威的 ADS 培训教程，共包含 10 门 ADS 学习培训课程。课程是由具有多年 ADS 使用经验的微波射频与通信系统设计领域资深专家讲解，并多结合设计实例，由浅入深、详细而又全面地讲解了 ADS 在微波射频电路设计、通信系统设计和电磁仿真设计方面的内容。能让您在最短的时间内学会使用 ADS，迅速提升个人技术能力，把 ADS 真正应用到实际研发工作中去，成为 ADS 设计专家...

课程网址： http://www.edatop.com/peixun/ads/13.html





HFSS 学习培训课程套装

该套课程套装包含了本站全部 HFSS 培训课程，是迄今国内最全面、最专业的 HFSS 培训教程套装，可以帮助您从零开始，全面深入学习 HFSS 的各项功能和在多个方面的工程应用。购买套装，更可超值赠送 3 个月免费学习答疑，随时解答您学习过程中遇到的棘手问题，让您的 HFSS 学习更加轻松顺畅…

课程网址：http://www.edatop.com/peixun/hfss/11.html

CST 学习培训课程套装

该培训套装由易迪拓培训联合微波 EDA 网共同推出，是最全面、系统、专业的 CST 微波工作室培训课程套装，所有课程都由经验丰富的专家授课，视频教学，可以帮助您从零开始，全面系统地学习 CST 微波工作的各项功能及其在微波射频、天线设计等领域的设计应用。且购买该套装，还可超值赠送 3 个月免费学习答疑…

课程网址：http://www.edatop.com/peixun/cst/24.html

HFSS 天线设计培训课程套装

套装包含 6 门视频课程和 1 本图书，课程从基础讲起，内容由浅入深，理论介绍和实际操作讲解相结合，全面系统的讲解了 HFSS 天线设计的全过程。是国内最全面、最专业的 HFSS 天线设计课程，可以帮助您快速学习掌握如何使用 HFSS 设计天线，让天线设计不再难…

课程网址：http://www.edatop.com/peixun/hfss/122.html

13.56MHz NFC/RFID 线圈天线设计培训课程套装

套装包含 4 门视频培训课程，培训将 13.56MHz 线圈天线设计原理和仿真设计实践相结合，全面系统地讲解了 13.56MHz 线圈天线的工作原理、设计方法、设计考量以及使用 HFSS 和 CST 仿真分析线圈天线的具体操作，同时还介绍了 13.56MHz 线圈天线匹配电路的设计和调试。通过该套课程的学习，可以帮助您快速学习掌握 13.56MHz 线圈天线及其匹配电路的原理、设计和调试…

详情浏览：http://www.edatop.com/peixun/antenna/116.html

我们的课程优势：

※ 成立于 2004 年，10 多年丰富的行业经验，

※ 一直致力并专注于微波射频和天线设计工程师的培养，更了解该行业对人才的要求

※ 经验丰富的一线资深工程师讲授，结合实际工程案例，直观、实用、易学

联系我们：

※ 易迪拓培训官网：http://www.edatop.com

※ 微波 EDA 网：http://www.mweda.com

※ 官方淘宝店：http://shop36920890.taobao.com