TRACE32 工具常用 CMM 脚本命令

项目名称	V4 项目
文档编号	
版 本 号	1.0.0
作　　者	TSS 组

文档更新记录

日期	更新人	版本	备注
2015-01-06	TSS课题组	1.0.0	创建

目 录

1. 信息显示区视图

1.1. 常用命令

1.1.1. AREA.Create

创建信息显示区，并设置其 size

1.1.2. AREA.RESET

移除所有信息显示区，设置"A000"为默认大小，所有的 print 输出和错误信息都在该信息显示区显示。这个命令不移除显示的信息窗口。

1.1.3. AREA.CLEAR

清空信息显示区

1.1.4. AREA.OPEN

打开信息显示区，并将内容输出到指定的文件中。

1.1.5. AREA.CLOSE

关闭文件的输出视图，如果没有给出信息区名称，则关闭所有的文件。

1.2. 举例

```
AREA.RESET
AREA.CREATE analysis 50. 50.      // analysis 表示信息区的名称
AREA.VIEW analysis
AREA.SELECT analysis
```

2. 变量声明

2.1. 本地变量声明

LOCAL &a &b &c　　//关健字为 LOCAL,变量以“&”开头
ENTRY &a &b
&c=&a*&b
RETURN &c

2.2. 全局变量声明

如：GLOBAL &State &Level　　//关健字为 GLOBAL,变量以“&”开头

3. 基本控制语句

3.1. 条件控制

基本的 IF-ELSE(注意 Trace32 关键字不区分大小写，也可以 if else)
如下：

```
IF "a"=="a"
(
    PRINT "true"
)
ELSE IF "a"=="b"
(
    PRINT "false"
)
ELSE
(
    PRINT "这里不会运行(test)"
)
```

Trace32 里面没有 then 关健字，多行语句请使用括号"()"括起来。

3.2. 循环语句

```
&true=0!=1
&count=1
```

```
WHILE &true
(
      DO mem_test
      PRINT "MEMTEST " &count
      &count=&count+1
)
ENDDO
```

```
 RePeaT [<count>]
  <block>
 [WHILE [<condition>]]
 or
 RePeaT <count> <command>
```

3.3. 跳转语句

```
GOTO endloop //关健字为 GOTO，endloop 为标号，如：
endloop:
     print "这里为 GOTO 执行地"

GOTO 102.    //102.为行号
```

4. 常用命令

4.1. 执行其他脚本

```
DO <filename> [<parlist>]
 比如有二个文件：
 a.cmm
 b.cmm
 a.cmm 内容为：do b
 b.cmm 内容为: print "a call b"
```

4.2. 调用子函数

```
GOSUB subr1 0x100 10. "abc"//调用子函数，关健字 GOSUB, subr1 为子函数标号，后面为调用的参数 0x100 10. "abc"
subr1:    //这里是子函数，以变量名加冒号，标号形式。
     ENTRY &address &len &string
     Data.Set &address++(&len-1) &string
```

RETURN

4.3. ENTER 和 ENTRY 语句

```
enter &x
 print "x=&x"

 ENTRY &address
 GOSUB func1 &address 1.
 ENTRY &result
 PRINT “Result=” &result
 ENDDO
 func1:
  LOCAL &addr &size
  ENTRY &addr &size
  Data.Set &addr++&size 0x0
  &retval=Data.Byte(&addr)
  RETURN &retval
```

4.4. INKEY 字符输入

```
 INKEY
 INKEY &key
 IF &key==0x0d
  print "正确的输入"
 else
  print "错误的字符"
```

4.5. data.load.coff

加载.a 文件，如：

```
data.load.coff "~~~~/phyfa.a"
```

4.6. DATA.LOAD.BINARY

加载现场数据，如：

```
DATA.LOAD.BINARY PL_MEM_X1643_DTCM_DATA.bin      D:0x0
```

4.7. WinPOS

产生一个窗口显示后面命令生成的内容。
格式:
WinPOS [<*pos*>] [<*size*>] [<*scale*>] [<*name*>] [**<***state*>] [<*header*>]
<*pos*>: **<***x-coordinates*> <*y-coordinates*>
<*size*>: **<***x-dimension*> <*y-dimension*> (integer only)
<*scale*>: **<***scale-width*> <*scale-height*> (integer only)
<*state*>: **Normal** | **Iconic | Maximized**
例：

```
WinPOS 1. 1. 20. 20. 2. ,, DUMP
Data.dump 0x1000
```

4.8. Data.List

显示 PC 指针附近的源码列表

4.9. REGISTER.VIEW

显示寄存器信息

4.10. DATA.DUMP

显示 dump 窗口

4.11. VAR.WATCH

打开变量查看窗口，并显示变量信息
格式:
Var.Watch[**%**<*format*>] [<*variable*>] …

4.12. Var.AddWatch

在变量查看窗口中添加变量
格式:
Var.AddWatch [**%**<*format*>] [<*variable*>] …

4.13. SYMBOL.BROWSE

打开符号浏览窗口

4.14. Data.Set

格式:
Data.Set [<*addressrange*>] **%**<*format*> <*string*> [*/*<*option*>]
<*format*>: **Byte | Word | Long | Quad | TByte | TWord**
Float. [Ieee | IeeeDbl | IeeeeXt | <*others*>]
BE | LE
<*option*>: **Verify | ComPare | DIFF | CORE** <*core*>

例：`Data.Set 0x100 %Long 0x12345678`

4.15. PRINT

打印字符串至信息显示区视图
格式:
PRINT [%<*format1*>] [%<*format2*>] <*data*>
<*format1*>: **Ascii**
BINary
Decimal
Hex
String
converts all following <data> to specified format
<*format2*> **ATTR <value>** print data with special attribute (internal feature only)
CONTinue print data to current instead of new line
ERROR data will be printed in red.
Only when printing to default area A000, the text is also
shown as error in the message line.
HOME print to top line in AREA window

5. 常用函数

5.1. VAR.VALUE

返回参数中表达式的值，例：

```
PRINT VAR.VALUE(g_st_error_info_save.b24_error_cause)
```

5.2. Var.STRing

以字符串格式返回参数中表达式的值，例：

```
&enumvalue=Var.STRing(ptr->member)
PRINT "&enumvalue"
```

5.3. Var.SIZEOF

返回表达式所占内存的大小，例：

```
Data.Print flags++Var.SIZEOF(flags)
```

5.4. Var.ADDRESS

返回表达式的地址，例：

```
Data.Print Var.ADDRESS(flags[3])
```

5.5. Data.Long

返回指定内存地址中的 32 位长整形值。如：

```
PRINT Data.Long(D:0x200)
```

类似的还有 Data.short, Data.Float, Data.Byte, Data.LongLong, Data.Short 等。