TYX PAWS
Release Notes - Miscelaneous
06/10/99 (19990610)

1.0 Overview

This document describes changes included within:

Module Name	Module Description
SysConf	System Configuration
TPS	TYX Programming Support
PORF	PAWS Output Report Formatter
Targetter	ATLAS Program Targetter
WinUtil	TYX MS Windows Utilities
PLI	PAWS LAPS Interpreter
PTE	PAWS Test Executive

Note SysConf is used by TPS, PORF, Targetter, WinUtil, PLI and PTE.
TPS is used by PORF, Targetter, WinUtil, PLI and PTE.
WinUtil is used by PLI and PTE.

Module	Changes	Rebuilt	Current Version
SysConf	Minor	N/A	19990610
TPS	Minor	Yes	19990610
PORF	Minor	Yes	19990610
Targetter	Minor	Yes	19990610 3.8.3
WinUtil	None	Yes	19990610
PLI	Minor	Yes	19990610 3.9.8
PTE	Major	Yes	19990610 3.9.12

PTE means both the Run-Time System (RTS) and the Simulator (SIM).

PTE, RTS, CEM and SIM mean all Platforms.
UPTE, URTS, UCEM and USIM mean UNIX Platforms only.
WPTE, WRTS, WCEM and WSIM mean MS-Windows Platforms only.
XPTE, XRTS and XSIM mean X-Window Platforms only.

N O T I C E TO C E M U S E R S

This version of the RTS is compatible with CEM Modules built with CEM Files distributed with RTS Version 19980714 3.9.7 and later. You may install this version of the RTS without rebuilding your CEM Module. However, in order to make use of the new CEM capabilities, you MUST install this version of the RTS, and then you MUST recompile and relink your CEM Module(s) with the TYX-supplied CEM Files distributed with this version of the RTS.

As time goes on, RTS/CEM testing to provide error-free backwards-compatibility becomes increasingly difficult. Therefore, TYX Corporation very strongly recommends that CEM Users recompile and relink

their CEM Modules after installing a new version of the RTS.

1.1 Enhancements

1.1.1 System Configuration – Minor.

Creation of new System SYS09D - SPARC SynOS 5.7+.

1.1.2 TYX Programming Support - Minor.

Support for new System SYS09D.

1.1.3 PAWS Output Report Formatter - Minor.

Support for new System SYS09D.

1.1.4 ATLAS Program Targetter - Minor.

Support for new System SYS09D.

1.1.5 TYX Windows Utilities - None.

1.1.6 PAWS LAPS Interpreter - Minor.

Support for new System SYS09D.

1.1.7 PAWS Test Executive - Major.

Support for new System 09D.
RTS-to-CEM Information Messages:
- Controller Address Information Message.

CEM Driver access to CEM Kernel Current State Information:
- Current Module Name.
- Current Line Number.
- Current Statement Number.
- Current Verb.

CEM Driver capability to determine Datum "freshness".
CEM User Stub Functions.

1.2 Problem Reports.- None.

2.0 Detailed Description

2.1 Enhancements

2.1.1 System Configuration - Minor

2.1.1.1 Creation of new System SYS09D - SPARC SynOS 5.7+.

Minor internal changes were made and new Files were created to support the new System.

2.1.2 TYX Programming Support - Minor.

2.1.2.1 Support for new System SYS09D.

Minor internal changes were made and new Files were created to support the new System.

2.1.3 PAWS Output Report Formatter - Minor.

2.1.3.1 Support for new System SYS09D.

Minor internal changes were made to support the new System.

2.1.4 ATLAS Program Targetter - Minor.

2.1.4.1 Support for new System SYS09D.

2.1.5 TYX Windows Utilities - None.

2.1.6 PAWS LAPS Interpreter - Minor.

2.1.6.1 Support for new System SYS09D.

Minor internal changes were made to support the new System.

2.1.7 PAWS Test Executive - Major.

2.1.7.1 Support for new System 09D.

Minor internal changes were made to support the new System.

2.1.7.2 RTS-to-CEM Information Messages.

2.1.7.2.1 Controller Address Information Message.

With the previous release, RTS and CEM were enhanced so that the RTS would provide the CEM Kernel with Device (but not Controller) Bus Addresses (MTA, MLA and MSA) prior to the first INTERFACE CLEAR / RESET / DEVICE CLEAR sequence.

With this release, the RTS provides the CEM Kernel with Controller Bus Addresses (MTA and MLA) (for the first "Channel" Bus specified in the Bus Configuration File) prior to the first INTERFACE CLEAR / RESET / DEVICE CLEAR sequence. The format of the RTS-to-CEM Controller Bus Addresses Information Message is:

`MI MTA:1stCemBus=`*ControllerMTA* **`MLA:1stCemBus=`***ControllerMLA*

For example:

```
MI MTA:1stCemBus=30 MLA:1stCemBus=30
```

Note Testing of this feature revealed that there are flaws in the design and implementation of the CEM Model.

The original (and still current) CEM Model design supported only a single "Channel" Bus. Today, however, it is not abnormal for a Test Station to have more than one Instrument Bus (e.g., two IEEE-488 Busses, 2 VXI Busses, or 1 IEEE-488 Bus and one VXI Bus) with all Instruments supported by CEM Instrument Drivers.

Sometime in the past, the CEM Model implementation was modified to allow the specification of multiple "Channel" Busses in the Bus Configuration File. Although these modifications provided limited support for multiple Instrument Busses, in some situations, they simply did not (and still do not) correctly support multiple Instrument Busses.

It is anticipated that multiple Instrument Busses will be correctly supported in the very near future (hopefully in the next RTS release).

2.1.7.3 CEM Driver access to CEM Kernel Current State Information.

With the previous release, the RTS and CEM were enhanced so that the RTS would provide the CEM Kernel with ATLAS Statement Information. However, that information was not available to CEM Drivers.

With this release, CEM Drivers have access to the following ATLAS Statement Information:

ATLAS Statement Information	CEM Macro
Current Module Name	*GetCurModName()*
Current Line Number	*GetCurLineNum()*
Current Statement Number	*GetCurStatNo()*
Current Verb	*GetCurVerb()*

Refer to Section 3 in this document for a full description of these CEM Macros.

2.1.7.4 CEM Driver capability to determine Datum "freshness".

Prior to this release, CEM Drivers had no way to determine Datum "freshness". This limitation caused CEM Drivers a problem when an ATLAS SETUP Statement specifying a Descriptor Modifier was directed to an active Instrument. For example, if an ATLAS APPLY Statement specifies POS-SLOPE and a subsequent ATLAS SETUP Statement specifies NEG-SLOPE, the CEM Driver Setup Function (called as a result of the ATLAS SETUP Statement) sees a Datum for each Modifier with no way to determine which one is "fresh" (i.e., which one was specified with the ATLAS SETUP Statement).

With this release, CEM Drivers can determine Datum "freshness" by calling the CEM Macros *IsOkDatumFresh()* and *IsDatumFresh()*. Refer to Section 3 in this document for a full description of these CEM Macros.

2.1.7.5 CEM User Stub Functions.

With this release, the CEM Header and Library Files support CEM User Stub Functions. Refer to Section 4 in this document for a full description of these Functions.

2.2 Problem Reports - None.

3.0 CEM Help

Update to Current State CEM Macro Group

GetCurLineNum()
GetCurModName()
GetCurStatNo()
GetCurVerb()
IsDatumFresh()
IsOkDatumFresh()

Name: GetCurLineNum

Type: Current State

Usage:
```
long  nLineNumber;
nLineNumber = GetCurLineNum();
```

Description: This Macro causes the Long Variable **`nLineNumber`** to be set to the Line Number (within the Current ATLAS Module Source File) of the Current ATLAS Statement .

If an error is detected or if the Current Line Number is not available, **`nLineNumber`** will be set to zero.

See Also: CEM Macro *GetCurModName()*.

Name: GetCurModName

Type: Current State

Usage:
```
char  *pcModuleName;
pcModuleName = GetCurModName();
```

Description: This Macro causes the Character Pointer **pcModuleName** to be set to a pointer to a Character Array which contains the Current Module Name (as specified in the BEGIN ATLAS Statement).

If an error is detected or if the Current Module Name is not available, **pcModuleName** will be set to a Null Pointer.

See Also: CEM Macros *GetCurLineNum()*, *GetCurStatNo()* and *GetCurVerb()*.

Name: GetCurStatNo

Type: Current State

Usage:
```
long  nStatementNumber;
nStatementNumber = GetCurStatNo();
```

Description: This Macro causes the Long Variable **nStatementNumber** to be set to the Statement Number (within the Current ATLAS Module) of the Current ATLAS Statement .

If an error is detected or if the Current Statement Number is not available, **nStatementNumber** will be set to a negative value (currently minus one (-1)).

See Also: CEM Macro *GetCurModName()*.

Name: GetCurVerb

Type: Current State

Usage:
```
short  nVerb;
nVerb = GetCurVerb();
```

Description: This Macro causes the Short Variable **nVerb** to be set to the Verb Mnemonic that represents the Current ATLAS Statement Verb.

Verb Mnemonics are of the form **V_XXX** (e.g., **V_APP** for “APPLY”) and may be found in the PAWS-generated CEM Header File **key.h**.

If an error is detected or if the Current Verb is not available, **nVerb** will be set to zero.

See Also: CEM Macro *GetCurModName()*.

Name: IsDatumFresh

Type: Current State

Usage:
```
struct DATUM  *psDatum;
psDatum = GetDatum( ... );
if( IsOkDatumFresh( psDatum ) )
{     if( IsDatumFresh( psDatum ) )
      {     Process Fresh Datum ...
      }
      else
      {     Process Stale Datum ...
      }
}
```

Description: This Macro returns: YES/TRUE (currently plus one (1)) if the specified Datum is "fresh"; or NO/FALSE (currently zero (0)) if the specified Datum is "stale".

Whenever the CEM Kernel receives a Function/Setup (FNC/SET) Message for a particular Device/Channel from the RTS, the CEM Kernel stores a unique number in the storage area reserved for that Device/Channel. This unique number is called the Device/Channel Function Identifier.

Whenever the CEM Kernel receives data from the RTS, the CEM Kernel stores a unique number in the storage area reserved for the Datum. This unique number is called the Datum Function Identifier.

If a Datum Function Identifier is greater than or equal to its Device/Channel Function Identifier, the Datum is defined to "belong" to the most recent FNC/SET Message for its Device/Channel and is therefore defined to be "fresh".

If a Datum Function Identifier is less than its Device/Channel Function Identifier, the Datum is defined to "not belong" to the most recent FNC/SET Message for its Device/Channel and is therefore defined to be "stale".

Consider the following example (assuming the Test Station has only one DC Power Supply):

```
REMOVE, DC SIGNAL, ... $
APPLY, DC SIGNAL, POS-SLOPE, VOLTAGE 10 V, DC-OFFSET 5 V, ... $
SETUP, DC SIGNAL, NEG-SLOPE, VOLTAGE 20 V, FREQ 400 HZ, ... $
```

When the CEM Driver SETUP Function is called to process the data associated with the APPLY Statement, the POS-SLOPE, VOLTAGE and DC-OFFSET Data will be "fresh" because they came along with the most recent (i.e., the APPLY) FNC/SET Message from the RTS.

When the CEM Driver SETUP Function is called to process the data associated with the SETUP Statement, the NEG-SLOPE, VOLTAGE and FREQ Data will be "fresh" because they came along with the most recent (i.e., the SETUP) FNC/SET Message from the RTS. But the POS-SLOPE and DC-OFFSET Data are still available because the Device/Channel has not been reset; these Data are now "stale" because they came along before the most recent FNC/SET Message.

If an error is detected or if it not possible to determine if the Datum is "fresh" or "stale", NO/FALSE is returned.

See Also: CEM Macro *IsOkDatumFresh()*.

Name: IsOkDatumFresh

Type: Current State

Usage:
```
struct DATUM  *psDatum;
psDatum = GetDatum( ... );
if( IsOkDatumFresh( psDatum ) )
{     if( IsDatumFresh( psDatum ) )
            ...
}
```

Description: This Macro returns: YES/TRUE (currently plus one (1)) if the CEM Macro *IsDatumFresh()* can determine if the specified Datum is "fresh" or "stale"; or NO/FALSE (currently zero (0)) if *IsDatumFresh()* cannot make such a determination.

The CEM Macro *IsDatumFresh()* determines if the specified Datum is either "fresh" or "stale", and returns either YES or NO accordingly. However, a third condition could exist: *IsDatumFresh()* may be unable to make the determination in the first place.

IsOkDatumFresh() should always be called prior to calling *IsDatumFresh()* to verify that it is okay to call *IsDatumFresh()* (i.e., that *IsDatumFresh()* can determine if the specified Datum is "fresh" or "stale").

See Also: CEM Macro *IsDatumFresh()*.

4.0 CEM User Stub Functions.

4.1 Overview

CEM User Stub Functions have been created in response to Customer requests and to TYX in-house testing requirements. The primary purpose of these Functions is to print CEM Kernel information (available to CEM Drivers) in a form that will assist the Programmer in writing a CEM Driver. Additionally, for those situations when the RTS is expecting a response from the CEM Module, the appropriate CEM User Stub Function will provide one.

These Functions also allow a Programmer to quickly create a dummy CEM Module. Testing with a dummy CEM Module is generally better then testing with simulated Busses and Devices because:

- It forces the RTS to execute its non-simulation code.
- The Operator does not have to enter simulated Device input (which works fine as long as the ATLAS Test Program does not attempt to "verify" the input).

  Note There is a small possibility that at some future time the response capability will be enhanced to remove this limitation.

To create an ATLAS Test Program and CEM Module that can be executed by the RTS, the Programmer must create, as a minimum, the following Files:

- ATLAS Test Program Source File.
- Device Database Source File.
- CEM C Language Source File.
- Device Database Definition File.
- Bus Configuration File.

To fully complete the package, the Programmer may also create the following Files:

- Switch Database Source File.
- Interface Adapter Source File.

All builds are then done normally, including the CEM Module build.

4.2 Enabling.

All CEM User Functions have the same syntax: they are "int" Functions that take no Arguments. The CEM User Stub Functions conform to this syntax and they all return zero (0). (The CEM Kernel completely ignores the return value.)

There are CEM User Stub Functions for all CEM Non-ATLAS and ATLAS Actions, and they may be enabled in one of two ways, either:

- By specifying a CEM User Stub Function in the Device Database Definition File (which will completely "stub out" that particular Action); or

- By directly calling a CEM User Stub Function from within an existing CEM User Function.

Note Programmers must make sure that a particular CEM User Stub Function is enabled only for the CEM Action for which it is designed. If a CEM User Stub Function is enabled for a CEM Action for which it is not designed, the least problem is that the displayed information will be erroneous, and the worst possible problem is that the CEM User Stub Function will cause a fault.

CEM User Stub Functions for Non-ATLAS Actions

Non-ATLAS Action	CEM User Stub Function
close	**userStubNaaCLOSE();**
connect	**userStubNaaCONNECT();**
deviceclear	**userStubNaaDEVICECLEAR();**
disconnect	**userStubNaaDISCONNECT();**
interfaceclear	**userStubNaaINTERFACECLEAR();**
interrupt	**userStubNaaINTERRUPT();**
open	**userStubNaaOPEN();**

CEM User Stub Functions for ATLAS Actions

ATLAS Action	CEM User Stub Function
close	**userStubCLOSE();**
connect	**userStubCONNECT();**
disconnect	**userStubDISCONNECT();**
fetch	**userStubFETCH();**
init	**userStubINIT();**
load	**userStubLOAD();**
open	**userStubOPEN();**
reset	**userStubRESET();**
setup	**userStubSETUP();**
status	**userStubSTATUS();**

Note The Functions specified for each Device/Channel's RESET ATLAS Action are also called immediately after the INTERFACECLEAR Non-ATLAS Action occurs and immediately before the DEVICECLEAR Non-ATLAS Action occurs.

4.3 Processing.

The bulk of the processing performed by the CEM User Stub Functions consists of acquiring information from the CEM Kernel, converting it to a form that is meaningful to a CEM Driver Programmer, and printing it. This processing is discussed in the Section on formatting below. However, there is some processing which is peculiar to the CEM User Stub Functions (i.e., something a CEM Driver would not do) and that processing is discussed in this Section.

userStubNaaINTERRUPT	Prints information only the first time it is called.
userStubCONNECT	Loops through and prints all Values of Connect-qualified Data.
userStubDISCONNECT	Loops through and prints all Values of Disconnect-qualified Data.
userStubFETCH	Loops through and stores a value in every Data Value of the Fetch-qualified Datum. (If the Datum Data Type is DIGITAL, a value is stored in every Data Value Byte.) Although these values are not strictly random (the first value stored is 1, the second is 2, the third is 3, etc.), from the standpoint of the ATLAS Test Program, they are effectively random. Loops through and prints all Values of the Fetch-qualified Datum.
userStubLOAD	Loops through and prints all Values of the Load-qualified Datum.
userStubSETUP	Loops through and prints all Values of all Data (except for Connect-, Disconnect-, Fetch- and Load-qualified Data) of all Modifiers for the Current Device/Channel.

4.4 Formatting.

CEM User Stub Functions print two types of Lines: Action Lines and Modifier Data Lines.

Every CEM User Stub Function prints an Action Line. Only the following ATLAS Action CEM Stub Functions print Modifier Data Lines: CONNECT, DISCONNECT, FETCH, LOAD and SETUP.

4.4.1 Formatting - Action Lines.

Action Lines have the following format:

CemStub hh:mm:ss *DevChan***/***FncId Statement Signal Action BusAddr*

where	is
CemStub	a Literal.
hh:mm:ss	the time of day.
DevChan	*DeviceName***:CH***ChannelNumber*
	where — is
	DeviceName — the Device Name as it appears in the Device Database Definition File.
	:CH — a Literal.
	ChannelNumber — the Channel (i.e., the Function) Number as it appears in the Device Database Definition File.
/	a Literal.
FncId	the Device/Channel Function Identifier.
Statement	**[***ModuleName***]***LineNumber***:***StatementNumber Verb*
	where — is
	[— a Literal.
	ModuleName — the ATLAS Module Name as it appears in the BEGIN Statement.
	] — a Literal.
	LineNumber — the ATLAS Statement Line Number in the ATLAS Source File.
	: — a Literal.
	StatementNumber — the six-digit ATLAS Statement Number.
	Verb — the ATLAS Statement Verb CIIL String.
Signal	*Noun* **(***MeasuredChar***)**
	where — is
	Noun — the ATLAS Statement Noun CIIL String.
	(— a Literal. Printed only if Measured Characteristic available.
	MeasuredChar — the ATLAS Statement Measured Characteristic CIIL String. Printed only if Measured Characteristic available.
	) — a Literal. Printed only if Measured Characteristic available.
Action	the Non-ATLAS or ATLAS Action.
BusAddr	**MTA=***MTA* **MLA=***MLA* **MSA=***MSA*
	where — is
	MTA= — a Literal.
	MTA — the IEEE-488 Bus My-Talk-Address of the Device.
	MLA= — a Literal.
	MLA — the IEEE-488 Bus My-Listen-Address of the Device.
	MSA= — a Literal.
	MSA — the IEEE-488 Bus My-Secondary-Address of the Device.

The following are examples of Non-ATLAS Action Lines:

```
CemStub 12:34:56 CLOSE

CemStub 12:34:56 CONNECT

CemStub 12:34:56 DISCONNECT

CemStub 12:34:56 DEVICECLEAR MTA=94 MLA=62
CemStub 12:34:56 DEVICECLEAR [atp]145:999999 TRM MTA=94 MLA=62

CemStub 12:34:56 INTERFACECLEAR
CemStub 12:34:56 INTERFACECLEAR [atp]145:999999 TRM

CemStub 12:34:56 INTERRUPT

CemStub 12:34:56 OPEN

CemStub 12:34:56 RESET acps:CH0 MTA=65 MLA=33 MSA=-1
```

Notes pertaining to the preceding examples

- *Statement* information shown with DEVICECLEAR and INTERFACECLEAR is printed only when these CEM User Stub Functions are called as a result of the execution of an ATLAS Statement (e.g., TERMINATE, ATLAS PROGRAM).

- *BusAddr* information shown with DEVICECLEAR contains the IEEE-488 Bus Addresses of the CEM Bus Controller. *BusAddr* information shown with RESET contains the IEEE-488 Bus Addresses of the corresponding Device.

- The RESET examples show the information printed when the RESET CEM User Stub Function is called as part of the INTERFACECLEAR / DEVICECLEAR Non-ATLAS Actions sequence.

The following are examples of ATLAS Action Lines for ATLAS SOURCE-type Statements:

```
CemStub 12:34:56 acps:CH0/5 [atp]49:101010 APP ACS CLOSE

CemStub 12:34:56 acps:CH0/5 [atp]49:101010 APP ACS CONNECT

CemStub 12:34:56 acps:CH0/9 [atp]99:104010 REM ACS DISCONNECT

CemStub 12:34:56 dwg:CH0/13 [atp]122:201550 DO DGT (STRC) LOAD Action=LOAD

CemStub 12:34:56 acps:CH0/9 [atp]99:104010 REM ACS OPEN

CemStub 12:34:56 acps:CH0/9 [atp]99:104010 REM ACS RESET MTA=65 MLA=33 MSA=-1

CemStub 12:34:56 acps:CH0/5 [atp]49:101010 APP ACS SETUP
CemStub 12:34:56 acps:CH0/9 [atp]91:103110 SET ACS SETUP

CemStub 12:34:56 acps:CH0/5 [atp]49:101010 APP ACS STATUS
```

Notes pertaining to the preceding examples

- ATLAS SOURCE-type Statements (e.g., APPLY) can be recognized by the absence of *MeasuredChar*.

- The LOAD ATLAS Action can occur for reasons other than the processing of "load" data. The LOAD CEM User Stub Function prints the reason: **LOAD**, **BEGIN** or **END**.

- *BusAddr* information shown with RESET contains the IEEE-488 Bus Addresses of the corresponding Device.

The following are examples of ATLAS Action Lines for ATLAS SENSOR-type Statements:

```
CemStub 12:34:56 dmm:CH2/6 [atp]60:102020 MEA ACS (CURR) CLOSE

CemStub 12:34:56 dmm:CH2/6 [atp]60:102020 MEA ACS (CURR) CONNECT

CemStub 12:34:56 dmm:CH2/6 [atp]60:102020 MEA ACS (CURR) DISCONNECT

CemStub 12:34:56 dmm:CH2/6 [atp]60:102020 MEA ACS (CURR) FETCH

CemStub 12:34:56 dmm:CH2/6 [atp]60:102020 MEA ACS (CURR) INIT

CemStub 12:34:56 dmm:CH2/6 [atp]60:102020 MEA ACS (CURR) OPEN

CemStub 12:34:56 dmm:CH2/6 [atp]60:102020 MEA ACS (CURR) RESET MTA=69 MLA=37 MSA=-1

CemStub 12:34:56 dmm:CH2/6 [atp]60:102020 MEA ACS (CURR) SETUP

CemStub 12:34:56 dmm:CH2/6 [atp]60:102020 MEA ACS (CURR) STATUS
```

Notes pertaining to the preceding examples

- ATLAS SENSOR-type Statements (e.g., MEASURE) can be recognized by the presence of *MeasuredChar*.

- *BusAddr* information shown with RESET contains the IEEE-488 Bus Addresses of the corresponding Device.

4.4.2 Formatting - Modifier Data Lines

Modifier Data Lines have the following format:

CemStub *EightSpaces ModifierInfo* { *DatumInfo ValueInfo* ... } ...

where	is
CemStub	a Literal.
EightSpaces	Eight (8) ASCII Space Characters (in place of the **hh:mm:ss** of Action Lines).
ModifierInfo	**M***m***:** *Modifier*

where	is
M	a Literal.
m	the Modifier Number. It starts at one (1) and is used for reference only; it has no meaning to CEM Drivers.
:	a Literal.
Modifier	the Modifier CIIL String.

DatumInfo	**D***d***/***f* **=Stale:** *Qualifier* **DS=***s*

where	is
D	a Literal.
d	the Datum Number. It starts at one (1) and is used for reference only; it has no meaning to CEM Drivers.
/	a Literal. Printed only if Datum Function Identifier not identical to most recent Device/Channel Function Identifier.
f	the Datum Function Identifier. Printed only if not identical to most recent Device/Channel Function Identifier.
=Stale	a Literal. Printed only if Datum Function Identifier less than most recent Device/Channel Function Identifier.
:	a Literal.
Qualifier	the Datum Qualifier CIIL String (e.g., **SET**, **SRX**, **FDD**).
DS=	a Literal. Printed only if Datum Data Type is DIGITAL.
s	the Size (in Bytes) of the Datum. Printed only in Data Type is DIGITAL.

ValueInfo	**V***v***:** *Value*

where	is
V	a Literal.
v	the Datum Value Index. It starts at zero (0) and is the index used by CEM Drivers to access a Datum Value.
:	a Literal.
Value	the Datum Value. Its format is dependent upon (and reflective of) the Datum Type.

The following are examples of ATLAS Statements, Action Lines and Modifier Data Lines:

APPLY, AC SIGNAL, VOLTAGE 10 V, FREQ RANGE 55 HZ TO 65 HZ, POS-SLOPE, ...

```
CemStub 12:34:56 acps:CH0/5 [atp]49:101010 APP ACS SETUP
CemStub          M1: POSS D1: SET
CemStub          M2: FREQ D1: SRN V0=+.55E+02 D2: SRX V0=+.65E+02
CemStub          M3: VOLT D1: SET V0=+.1E+02
```

Note For each Modifier, there will always be at least one Datum. However, a Datum may have no Values (as is the case with M1-D1).

SETUP, AC SIGNAL, FREQ 1000 HZ, NEG-SLOPE, ...

```
CemStub 12:34:56 acps:CH0/9 [atp]91:103110 SET ACS SETUP
CemStub          M1: NEGS D1: SET
CemStub          M2: POSS D1/5=Stale: SET
CemStub          M3: FREQ D1/5=Stale: SRN V0=+.55E+02 D2/5=Stale: SRX
CemStub              V0=+.65E+02 D3: SET V0=+.4E+03
CemStub          M4: VOLT D1/5=Stale: SET V0=+.1E+02
```

Note M1-D1 and M3-D3 are “fresh” because they are a result of the ATLAS SETUP Statement. All others are “stale” because they are a result of the previous ATLAS APPLY Statement.

MEASURE, AC SIGNAL (CURRENT), ... CNX ...

```
CemStub 12:34:56 dmm:CH2/11 [atp]100:202020 MEA ACS (CURR) CONNECT
CemStub          M1: PATH D1: CON V0=1 V1=2 V2=2

CemStub 12:34:56 dmm:CH2/11 [atp]100:202020 MEA ACS (CURR) FETCH
CemStub          M1: CURR D1: FTH V0=+.1E+01
```

Note M1-D1-V0 is a result of the FETCH CEM User Stub Function processing and is the value that will be returned to the RTS.

DO, DIGITAL, ...

```
CemStub 12:34:56 dwg:CH0/13 [atp]122:301550 DO DGT (STRC) LOAD Action=LOAD
CemStub          M1: STIM D1: LOD DS=8 V0=x0123456789ABCDEF
CemStub              V1=x0123456789ABCDEF V2=x0123456789ABCDEF
```

4.5 Printing.

For the purpose of this Document, "printing" means the output of data for viewing by the Operator.

The CEM User Stub Functions will print data only if the Environment Variable **TYXCEMSTUBLEVEL** is set to a number greater than zero. If **TYXCEMSTUBLEVEL** is either undefined or is equal to zero, no data will be printed; however, the CEM User Stub Functions will continue to execute (including responding to RTS requests for data). The value of the number determines how much data will be printed:

TYXCEMSTUBLEVEL	Additional Data Printed
0	No data printed.
1	All data printed.

Note Currently, any number greater than or equal to one (1) will cause all data to be printed. This may be changed in the future so that higher numbers will cause more data to be printed. For the time being, Programmers should set the number to nine (9).

Printed data may be viewed by the Operator in one of three places:

RTS Operator Display The worst place to view the data due to the potential volume. No special action is required by the Operator to view this data; it is displayed automatically by the RTS whenever the CEM Module sends it.

RTS Log File (default File Name "LOG") A much better place to view the data. However, due to RTS / CEM interfaces, not all data is available, specifically the data for the CONNECT and DISCONNECT Non-ATLAS Actions. The Operator must enable the RTS Log File capability each time the RTS is activated.

CEM Log File (File Name "CEM.log") The best place to view the data. The Operator must enable the CEM Log File capability by setting the Environment Variable **TYXCEMFDEBUG** to a number greater then or equal to one (1). (A setting of two (2) is recommended to enable the printing of RTS-to-CEM Information Messages.) The Operator should also set the Environment Variable **TYXPATHLOG** to something like "C:\TMP" or "C:\LogFiles" - it doesn't matter what as long as its a convenient Directory or Sub-Directory to access. (If **TYXPATHLOG** is not defined, the CEM Log File will be created in the Current Working Directory. But since the RTS requests directory changes, it is sometimes troublesome to locate this File.)