HarnesSys™

User Manual
For
Import Libraries

Revision D_1

04 November 2010

Notices

The information contained in this manual is believed to be accurate and reliable. However, Israel Aerospace Industries Ltd. (IAI) assumes no responsibility for any errors, omissions, or inaccuracies whatsoever.

Information in this document is subject to change without notice and does not represent a commitment on the part of Israel Aircraft Industries Ltd. (IAI). The software and/or database described in this document are furnished under a license and/or agreement. The software and/or database may be used or copied only in accordance with the terms of such license and/or agreement. It is against the law to copy the software on any medium except as specifically allowed in the license and/or nondisclosure agreement. The purchaser may make one copy of the software for backup purposes. No part of this document and/or data base may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or information storage and retrieval systems, for any purpose other than the purchaser's personal use, without the express written permission of IAI.

HarnesSys™

User Manual For Import Libraries

Revision D_1

04 November 2010

	Name	Signature	Date
Prepared By:	Esther Itzhak		
Revised By:	J.P. Stroweis		04-NOV-2010
Checked By:	A. Bashkin		
Approved By:	S. Shiff		

Records of Changes and Revisions

Change No.	Revision	HarnesSYS Release	Date Issued	Description	Authors
1	NEW	R17a.00	07 May 2007	Original Manual	E. Yitzhak
2	B	R18a.00	25 Sep. 2008	Clarifications	J.P. Stroweis
3	C	R18a.06	04 Feb. 2009	Major update. New options on contact keys. New or enhanced description of the library importer approach, the precedence rules and the tracking tools. Extended information on the valid values for each field.	J.P. Stroweis
4	D	R19a.00	16 Dec. 2009	New keyword for contacts: STANDARD_CONTACT	J.P. Stroweis
5	D_1	R19a.19	04 Nov. 2010	Contact key refers to pin arrangement; Update list of library tables not in scope.	J.P. Stroweis

Table of Contents

List of Tables

General

Introduction

This user guide describes the instruction for importation of the Part/Wire/Termination code/ Box Signal Libraries from a legacy system into the ***HarnesSys***.

The user guide is applicable to ***HarnesSys*** version r18a.06 and above.

Audience

This manual is written for those ***HarnesSys*** users that are in charge of the importation of foreign wire harness information into a ***HarnesSys*** database project (typically, information system personnel, database administrator and legacy system administrator). It assumes a comprehensive grasp of the ***HarnesSys*** concepts, basic familiarity with UNIX and ability to understand the nature and content of the information to be imported.

Scope

This importer could be used for Part Spec, Wire Spec, Termination Code and Box Signal libraries.

Organization

This document is organized in several major sections, as follows:

- Introduction to the Importer, its goal, scope, and principles,
- The Part/Wire/Termination code/ Box Signal file data structure and sample,
- A presentation of the Importer user interface, with the tracking tools..

Disclaimer

The goal of the import is to allow the reuse of the legacy data in ***HarnesSys*** as if HarnesSys had originally created it. However, full reuse depends on the completeness and exactitude of the import data in the ***HarnesSys*** perspective. Partial import may only allow for partial reusability.

Conventions

Character Set

Characters shall belong to the 7-bit ASCII character set, including A-Z upper case and a-z lower case English letters, 0-9 digits, the space character and a series of punctuation characters and control characters.

Special Characters

Special characters are:
' " ` & * ( ) [ ] { } , . ; ^ @ etc.

Before a Special character enter the sign: ^

For example: for ABC(123 456) type ABC^(123^ 456^)
for ABC^DFG type ABC^^DFG

Syntax

The file syntax is made to be human readable, free-format and self explanatory.

The syntax is free format, in the sense that it may include any number of space or tabs characters between field values, and that the information can span over several consecutive lines. Leading and trailing space characters are ignored.

Typically, a record is made of one or more lines and ends with a semicolon (“;”) character.

A record starts by a record type keyword followed by an equal sign followed by a left parenthesis followed by list of one or more lists of "KEYWORD = value" assignments separated by commas (","). Keywords are fixed strings of upper case characters. The list has one or more lines.

If a field does not contain data, it may be omitted.

The program will issue an error message and the problematic part/terminator/wire spec/contact key will be rejected.

Mandatory and Optional fields

Some of the fields in the transaction file are mandatory (must) and some are optional. Some fields are mandatory in certain context only. A default value may be given to optional fields that are absent. All this information is provided for each type of transaction type.

Field information

For each specific transaction file document, a table describes the keywords and their values. Such a table has the following columns:

Column	Description
Keyword List	A keyword whose field type is a list. The authorized keywords in this list are then presented in the following rows of the table.
Keyword	The field keyword (upper case only); Each keyword is declared within the context of its keyword list.
Type	The field type within the transaction file (e.g. number, string)
Must	If marked "YES", this keyword is mandatory in the transaction file record, when its list is present. Otherwise, i.e. marked N (No) or cell left blank, it is optional. The mark N is placed to stress that the field is not mandatory. Interpretation of "Must" is context-dependent, as explained in the note below. Mandatory fields are written in **bold** characters in the description tables and in the samples.
Remarks	Additional useful information, including: • default value (if any), • location of default value in project defaults (if any), • is this field governed by format control, • expected character length, • location of default string length in project defaults (if any),
ICD Table	Principal target Oracle table, as defined in the Interface Control Document (ICD). The input value may be placed into several tables.

Figure 1: Keyword information in transaction file specifications

Units of Measurements

HarnesSys stores all data using a fixed set of units: mm, g and C degrees, which are translated to the required values during the report generation.

The import file will contain measurements with or without units:

- If the value comes with its units then they should be written as one string, without blank, e.g. 25g, 0.3mi.
- If the units are not specified then HarnesSys will understand that the measurement is provided is in **MKS** system (as stored in the Data Base).

The following units are allowed in the import data files:

Measurement	Unit	Comment
Length	m	meters
	cm	Centimeters
	mm	Millimeters
	in	Inches
	ft	Feet
	mi	Miles
	um	Micrometers, i.e. 0.001mm
	nm	Nanometers, i.e. 0.000001mm
Weight	kg	Kilograms
	g	grams
	lb	Pounds
	oz	Ounces
Weight per unit of length	g/m	
	kg/km	
	Lb/1000ft	
Resistance per unit of length	Ohms/km	
	Ohm/km	
	Ohms/mi	
	Ohms/1000ft	
Resistance ("Impedance" in HarnesSYS)	Ohms	
	mohm	0.001 Ohms
	Mohm	1000000 Ohms
Temperature	C	Celsius degrees
	F	Fahrenheit degrees

Figure 2: Units of Measurements

Importer Scope

Library tables are those database tables designated to serve as a central data source. Libraries do not contain any specific electrical information about documents, wires or parts used in a given vehicle project. They are referenced by the documents, wires or parts of the project.

It is recommended to populate them before the first documents, wires or parts are created, but this is not a must.

This tool supports the import of most of the libraries from Legacy systems that are used in an electrical project.

If a given functionality does not exist in your Legacy system, you may decide not to populate this group/table during conversion. Missing library data can be added at a later stage.

Please note that the import utility does not cover all the libraries:

1. The following library tables are pre-populated during at create project time (but it is possible to modify them interactively):

   - GAUGE_CMA
   - WIRE_ACCOMODATION

2. The following library tables are pre-populated during at create project time and they should not be modified by the customer:

   - ERROR_DESCRIPTION
   - ERROR_MESSAGE_CODES

3. The following library tables usually hold such a small amount of information, that it is as simple to fill them with HarnesSYS interactively:

   - AREA_CODE
   - BRAID
   - EMI_RULES

4. Some library tables were used in the past, to support the "old terminator" functionality. It is now strongly recommended to use the "new terminator" functionality, which uses a different set of tables (supported by this importer). When using the "new terminator" functionality, just ignore the following tables:

   - BINCODE
   - BINCODE_AWG
   - BINCODE_RATING
   - BINCODE_TOOL
   - BINGROUP
   - CONTACT
   - PINBINCODE
   - TOOL

5. The following tables are not used now:

   - BUS
   - EXPERT_WIRES2LUG
   - PART_SPEC_SYMBOL
   - PART_SPEC_SYMBOL_CONTACT
   - POWER_SOURCE
   - RATING

6. The following tables have no import mechanism so far:

   - GLOB_PIN_ARRANGE_MAP
   - GLOB_PIN_ARRANGE_MIL_STND
   - GLOB_PIN_ARRANGE_PIC
   - MFG_PROCESS
   - PART_SPEC_LOAD
   - PART_SPEC_ASSEMBLY
   - TERM_CODE_MULT_USAGES
   - TOOLS

For more details about the HarnesSYS database tables, please refer the latest ICD manual.

Precedence Rules

In order to avoid inconsistent data conflicts, the Connector Standards (MIL_STDS records) must be imported in a separate file prior to any other imports.

If during import of entities that refer to the Connector Standards (wire specs, terminators, contact keys) the program detects a MIL_USAGE reference to a connector standard/series, which is not listed in the Connector Standards, the program will issue an error message and the problematic part/terminator/wire spec/contact key will be rejected.

Then, the rest of the library entities can be imported as one single file or as a series of files.

In the case of a single file, it will include all the library entities, except the Connector Standards. The system will automatically sort the entities in this file to properly import everything.

In the case of a series of import files, it is the responsibility of the users to submit the files in a coherent order, to satisfy the references: tables that are referenced shall be populated first (e.g. VENDOR shall be imported prior to PART_SPEC, because PART_SPEC contains references to VENDOR). If a reference to a non-existing data is found, the item is rejected (e.g. if there is a reference from a PART_SPEC to a non-existing VENDOR, then the given PART_SPEC is rejected and an error message is issued).

Import Sequence

Find here an example of a global import sequence is:

#	Subjects	Record Types
1	Connector Standards (must be in a separate file)	MIL_STDS
2	Vendor	VENDOR
3	Term Codes (with their connector standards and related alternative part numbers)	TERM_CODES MIL_USAGE (when OBJECT_TYPE = TERM) ALTERN_TERM_PN COMBO_TERM_PN
4	Contacts (with their connector standards)	CONTACTS MIL_USAGE (when OBJECT_TYPE = CONTACT)
5	Part Description	DESC
6	Shell	SHELL
7	PPN	PPN
8	Connection Key	CONTACT_CONNECTION
9	Part Specifications (with their related connector standards)	PART_SPEC MIL_USAGE (when OBJECT_TYPE = SPEC)
10	Mating Part Numbers	MATING
11	Wire Specifications	WIRE_SPEC
12	Wire Specification – Colors	WIRE_SPEC_COLOR
13	Wire Specification – Cable	CABLE
14	Box Interfaces	BOX_INTERFACES
15	Box Signals	BOX_SIGNAL

Figure 3 Suggested Import Sequence

Connector Standards

MIL Standards

These records define the connector standards.

Record type - MIL_STDS
ICD Table - MIL_STDS

Keyword	Type	Must	Remarks
MIL_STND	string	YES	MIL standard
MIL_SER	string		MIL series
SUBST_MIL_STND	string		
SUBST_MIL_SER	string		
NEW_DES_SUBST_DATE	date		
GLOBAL_SUBST_DATE	date		
NOTE	string		Note is used for specific series
DESCRIPTION	string		Description is used to describe all series of specific connector standard

Sample

```
MIL_STDS = (
        MIL_STND                = <value>,
        MIL_SER                 = <value>,
        SUBST_MIL_STND          = <value>,
        SUBST_MIL_SER           = <value>,
        NEW_DES_SUBST_DATE      = <value>,
        GLOBAL_SUBST_DATE       = <value>,
        NOTE                    = <value>,
        DESCRIPTION             = <value>
);
```

Part Libraries

The following tables support the libraries of parts:

1. VENDOR
2. PART_SPEC
3. PART_DESCRIPTION
4. MATING_CONNECTOR
5. SHELL
6. PPN
7. BOX_INTERFACES
8. CONTACT_CONNECTION
9. MIL_USAGE
10. CONT_KEY_GROUPS
11. CONT_KEY_GROUP_CONTACTS
12. CONT_KEY_GROUP_OPTIONS

Vendor

These records serve to import vendor data. Vendor data is references by part specifications and wire specifications.

Record type – VENDOR
ICD Table - VENDOR

Keyword	Type	Must	Remarks
VENDOR_CODE	string	YES	
NAME	string	YES	
PHONE	string		
FAX	string		
TELEX	string		

Sample

VENDOR= (
VENDOR_CODE = <value>,
NAME = <value>,
PHONE = <value>,
FAX = <value>,
TELEX = <value>
);

Part Specification

These records serve to import the specifications of electrical parts. It aggregates intrinsic part properties with references to specific properties located in several tables (vendors, connections, contacts, shells, descriptions, PPNs).

Record type – PART_SPEC
ICD Table - PART_SPEC

Keyword	Type	Must	Remarks	
PART_NUMBER	string	YES		
DESCRIPTION	string	YES	If this value does not pre-exist in the DESCRIPTION table, then it will be added there.	
VENDOR_CODE	string		Reference to VENDOR_CODE in the VENDOR table	
SPEC_LCD	string		If the value is not provided, then the current date is used	
INTERNAL_NUMBER	string			
DWG_NUMBER	string			
WEIGHT_INDEX	string		**Possible Values**	**Meaning**
			E	Estimated
			D	Data Source
			C	Calculated
			W	Weighed
WEIGHT	number			
COOLING	number			
COOL_TYPE	string			
PART_TYPE	string		**Possible Values**	**Meaning**
			B	ModBlock
			BCK	Backshell
			BOX	Box
			BS	Bus Bar
			CB	Circuit Breaker
			J	Receptacle
			JCX	Coax Receptacle
			JTX	Triax Receptacle
			P	Plug
			PCX	Coax Plug
			PTX	Triax Plug
			RL	Relay without diode
			RLD	Relay with diode
			RLP	Relay with press to test
			UND	Undefined
BASE_PLUG_PROG_NO	string		Reference to the MAJOR field in the PPN table	
ACDC	string			
VOLT	number			
LOAD	number			
HZ	number			
CONTACT_KEY	string		Reference to the CONTACT_KEY_NAME field in the CONT_KEY_GROUPS table	

Keyword	Type	Must	Remarks	
CONTACT_TYPE	string		"PIN" or "SOCKET"	
CONNECTION_KEY	number		Reference to the CONNECTION_KEY field in the CONTACT_CONNECTION table	
SHELL_SIZE	string		Reference to the SHELL_SIZE field in the SHELL table	
TEST	string		**Possible Values**	**Meaning (as in GUI)**
			TEST	TEST
			MILITARY	MILITARY
			CIVIL	CIVIL
			SUPRSED	SUPERSEDE
			GRN_EQP	GROUND EQP.
TETRA_ETCH	string			
PART_CODE	string			
MIL_STND	string		In case of single association goes to MIL_USAGE	
MIL_SER	string		In case of single association goes to MIL_USAGE	
TEMPERATURE_FROM	number			
TEMPERATURE_TO	number			
PART_LENGTH	number			
PART_WIDTH	number			
PART_HEIGHT	number			
POWER_OUTPUT	number			
ICD_VCD	string			
WEIGHT_UNIT	string		"WEIGHT" (default) or "WEIGHT/LENGTH"	

Sample

PART SPEC = (
PART_NUMBER = **<value>,**
DESCRIPTION = **<value>,**
VENDOR_CODE = <value>,
SPEC_LCD = <value>,
INTERNAL_NUMBER = <value>,
DWG_NUMBER = <value>,
WEIGHT_INDEX = <value>,
WEIGHT = <value>,
COOLING = <value>,
COOL_TYPE = <value>,
PART_TYPE = <value>,
BASE_PLUG_PROG_NO = <value>,
ACDC = <value>,
VOLT = <value>,
LOAD = <value>,
HZ = <value>,
CONTACT_KEY = <value>,
CONTACT_TYPE = <value>,
CONNECTION_KEY = <value>,
SHELL_SIZE = <value>,
TEST = <value>,

```
   TETRA_ETCH              =   <value>,
   PART_CODE               =   <value>,
   MIL_STND                =   <value>,
   MIL_SER                 =   <value>,
   TEMPERATURE_FROM        =   <value>,
   TEMPERATURE_TO          =   <value>,
   PART_LENGTH             =   <value>,
   PART_WIDTH              =   <value>,
   PART_HEIGHT             =   <value>,
   POWER_OUTPUT            =   <value>,
   ICD_VCD                 =   <value>,
   WEIGHT_UNIT             =   <value>
);
```

Part Description

These records serve to import the description and the group of a part.

Record type - DESC
ICD Table - PART_DESCRIPTION

Keyword	Type	Must	Remarks
DESCRIPTION	string	YES	
PGROUP	string		
ABBREVIATION	string		
CONNECTION_KEY_REQUIRED	integer		1- YES, 0 or NULL – NO

Sample

```
DESC = (
   DESCRIPTION             =   <value>,
   PGROUP                  =   <value>,
   ABBREVIATION            =   <value>,
   CONNECTION_KEY_REQUIRED =   <value>
);
```

Mating Connector

These records serve to declare pairs of mating part numbers defined as part specifications.

Record type - MATING
ICD Table - MATING_CONNECTOR

Keyword	Type	Must	Remarks
PART1	string	YES	First part number
PART2	string	YES	Second part number

Sample

```
MATING      =      (
```

PART1 = <value>,
PART2 = <value>
);

Shell

These records serve to import the data relevant to the shell of connectors.

Record type - SHELL
ICD Table - SHELL

Keyword	Type	Must	Remarks
SHELL_SIZE	string	YES	
STRAIGHT_RECEPTACLE	string		
ANGLED_RECEPTACLE	string		
STRAIGHT_PLUG	string		
ANGLED_PLUG	string		

Sample

SHELL= (
SHELL_SIZE= <value>,
STRAIGHT_RECEPTACLE= <value>,
ANGLED_RECEPTACLE= <value>,
STRAIGHT_PLUG= <value>,
ANGLED_PLUG= <value>);

PPN

These records serve to import the plug programming number.

Record type - PPN
ICD Table - PPN

Keyword	Type	Must	Remarks
MAJOR	string	YES	
SUB	string		
DEGREE	number		
PIN_NAME	string	YES	
SEQ	number		
LAST_IN_ROW	string		

Sample

PPN= (
MAJOR= <value>,
SUB= <value>,

DEGREE= <value>,
PIN_NAME= <value>,
SEQ= <value>,
LAST_IN_ROW= <value>);

Contact Connection

These records serve to declare the internal electrical connections within a part.

Record type - CONTACT_CONNECTION
ICD Table - CONTACT_CONNECTION

Keyword	Type	Must	Remarks
CONNECTION_KEY	number	YES	
DEVICE_TYPE	string	YES	
PIN_IN	string	YES	
PIN_OUT	string	YES	
WORK_MODE	string	YES	
DC_VALUE	number		
DC_UNITS	string		
AC_VALUE	number		
AC_UNITS	string		
V_DROP	number		
TRIGGER_VALUE	number		
TRIGGER_UNITS	string		
SELF_LOCK	string		
ACTIVATE_IN	string		
ACTIVATE_OUT	string		

Sample

CONTACT_CONNECTION= (
CONNECTION_KEY = <value>,
DEVICE_TYPE = <value>,
PIN_IN = <value>,
PIN_OUT = <value>,
WORK_MODE = <value>,
DC_VALUE = <value>,
DC_UNITS = <value>,
AC_VALUE = <value>,
AC_UNITS = <value>,
V_DROP = <value>,
TRIGGER_VALUE = <value>,
TRIGGER_UNITS = <value>,
SELF_LOCK = <value>,
ACTIVATE_IN = <value>,
ACTIVATE_OUT = <value>
);

Contacts

These records serve to import sets of contacts called "contact keys".

Record type - CONTACTS
ICD Tables - MIL_USAGE
CONT_KEY_GROUPS
CONT_KEY_GROUP_CONTACTS
CONT_KEY_GROUP_OPTIONS

Starting from release HarnesSYS r18a.06, there are two alternatives for insert arrangements:

- The field CONTACT_NAME contains single contacts,
- The field CONTACT_NAME declares an insert arrangement formula. The formula is automatically converted by the importer into a list of separate contacts. The formula may contain commas and parentheses following the same rules of the interactive form. Each comma and each parentheses must be preceded by the ^ symbol (in order to distinguish them from the delimiters of this file).

Starting from release HarnesSYS r18a.06, there are two alternatives for terminations:

- Manual termination codes: import the contact keys with the termination codes known on the legacy system,
- Automatic termination codes: import the contact keys without termination codes. In this case, HarnesSYS will try to associate existing termination codes (already in the database, for example previously imported by keyword TERM_CODES) to the contacts of the contact key. Together with the termination codes, the wire barrel or the wire part number, the usage and the contact type are copied to the table CONT_KEY_GROUP_OPTIONS. This option is available only for PLUG contact style in the context of a known MIL_USAGE.

The benefits of the automatic method are (1) to support conversion of contact keys for legacy systems that do not associate termination codes to contact keys and (2) to spare from the user the need to declare these associations interactively after contact keys are imported.

The drawback of this method is that the automatic assignment of termination codes, done according to standards and gauge, may not be unambiguous since several candidate termination codes may answer the criteria (in which case the importer messages).

The benefit of the manual method is to reproduce in HarnesSYS exactly the conditions from the legacy system. In this case, the importer does not verify the termination codes.

In order to populate automatically the termination codes to be used on an insert arrangement of a given contact key, the importer needs the contacts definition as well as the mil_usage of this contact key. Therefore it is recommended to declare CONTACTS and MIL_USAGE keywords in the same import file. See the example below.

Manual method

In the case where termination codes for these contacts are imported "manually" from a legacy system, the record information is as follows:

Keyword	Type	Must	Remarks
CONTACT_KEY_NAME	string	YES	
CONTACT_STYLE	string	YES	"PLUG", "STUD" or "OTHER"
INSTALLATION_TYPE	string		"CRIMP", "SOLDER" or "OTHER"
PIN_ARRANGE_NAME	string		Refers to the correspondent pin arrangement identifier, which describes the plug chart structure.
MIL_STND	string		In case of single association
MIL_SER	string		In case of single association
AWG	string	YES	
CONTACT_NAME	string	YES	A single contact (e.g. A) or insert arrangement formula, e.g. A-Z^(I^,O^,Q^)^, 1-10^, BB
CONTACT_TYPE	string		"PIN" or "SOCKET" (relevant when CONTACT_STYLE = "PLUG")
WIRE_BARREL	number		The field may be filled automatically during terminators match for plugs.
WIRE_PART_NUMBER	string		The field may be filled automatically during terminators match for plugs.
USAGE	string		From CODE_DETAIL table where code='USAGE'
			Possible Values / **Meaning**
			PWR – POWER
			CX – COAX
			TX – TRIAX
			TW – TWINAX
			FB – FIBRE OPTIC
			QDRX – QUADRAX
			TC-AL – Alumel
			TC-CN – Constantan
			TC-CR – Chromel
			TC-CU – Copper
			TC-FE – Iron
NUMBER_OF_WIRES	number		
MANUAL_TERM_CODE	string		
MIN_CMA	number		Relevant for STUD only
MAX_CMA	number		Relevant for STUD only
MIN_WIRE_AWG	string		Relevant for STUD only
MAX_WIRE_AWG	string		Relevant for STUD only
STANDARD_CONTACT	string		Is the contact provided together with the part or not?
			Possible Values / **Meaning**

Keyword	Type	Must	Remarks	
			Y	contact supplied with part
			N	contact is not supplied w/ part
			per default	Y if contact style is set to PLUG"; otherwise N.

Sample for manual method

CONTACTS = (

CONTACT_KEY_NAME = <value>,
CONTACT_STYLE = <value>,
INSTALLATION_TYPE = <value>,
PIN_ARRANGE_NAME = <value>,

MIL_STND = <value>,
MIL_SER = <value>,

CONTACT_NAME = <value>,
AWG = <value>,
WIRE_BARREL = <value>,
WIRE_PART_NUMBER = <value>,
USAGE = <value>,
NUMBER_OF_WIRES = <value>,
MANUAL_TERM_CODE = <value>,
MIN_CMA = <value>,
MAX_CMA = <value>,
MIN_WIRE_AWG = <value>,
MAX_WIRE_AWG = <value>,
STANDARD_CONTACT = <value>,

First insert arrangement

CONTACT_NAME = <value>,
AWG = <value>,
WIRE_BARREL = <value>,
WIRE_PART_NUMBER = <value>,
USAGE = <value>,
NUMBER_OF_WIRES = <value>,
MANUAL_TERM_CODE = <value>,
MIN_CMA = <value>,
MAX_CMA = <value>,
MIN_WIRE_AWG = <value>,
MAX_WIRE_AWG = <value>,
STANDARD_CONTACT = <value>

Second insert arrangement

);

Automatic method

In the case where termination codes for these contacts are deducted automatically, the record information is as follows:

Keyword	Type	Must	Remarks	
CONTACT_KEY_NAME	string	YES		
CONTACT_STYLE	string	YES	"PLUG", "STUD" or "OTHER"	
INSTALLATION_TYPE	string		"CRIMP", "SOLDER" or "OTHER"	
PIN_ARRANGE_NAME	string		Refers to the correspondent pin arrangement identifier, which describes the plug chart structure.	
MIL_STND	string		In case of single association	
MIL_SER	string		In case of single association	
AWG	string	YES		
CONTACT_NAME	string	YES	A single contact (e.g. A) or insert arrangement formula, e.g. A-Z^(I^,O^,Q^)^, 1-10^, BB	
NUMBER_OF_WIRES	number			
MIN_CMA	number		Relevant for STUD only	
MAX_CMA	number		Relevant for STUD only	
MIN_WIRE_AWG	string		Relevant for STUD only	
MAX_WIRE_AWG	string		Relevant for STUD only	
STANDARD_CONTACT	string		Is the contact provided together with the part or not?	
			Possible Values	**Meaning**
			Y	contact supplied with part
			N	contact is not supplied w/ part
			per default	Y if contact style is set to PLUG"; otherwise N.

Sample for the automatic method

CONTACTS= (

CONTACT_KEY_NAME = <value>,
CONTACT_STYLE = <value>,
INSTALLATION_TYPE = <value>,
PIN_ARRANGE_NAME = <value>,

MIL_STND = <value>,
MIL_SER = <value>,

CONTACT_NAME = <value>,
AWG = <value>,
STANDARD_CONTACT = <value>,

} First insert arrangement

```
    CONTACT_NAME        = <value>,
    AWG                 = <value>,
    STANDARD_CONTACT    = <value>
);
```

Second insert arrangement

Example for the automatic method

As a pre-requisite to automatic assignment, let us assume that termination codes were previously imported via the following lines:

```
TERM_CODE = (
    TERM_CODE       = PIN_1,
    PART_NUMBER     = PIN_1_P_N,
    TYPE            = PIN,
    AWG             = 20,
    WIRE_BARREL     = 20,
    USAGE           = PWR
);
MIL_USAGE = (
      MIL_STND      = 38999,
      MIL_SER       = I,
      OBJECT_ID     = PIN_1,
      OBJECT_TYPE   = TERM
);
MIL_USAGE = (
      MIL_STND      = 38999,
      MIL_SER       = II,
      OBJECT_ID     = PIN_1,
      OBJECT_TYPE   = TERM
);
TERM_CODE = (
      TERM_CODE     = PIN_2,
      PART_NUMBER   = PIN_2_P_N,
      TYPE          = PIN,
      AWG           = 22,
      WIRE_BARREL   = 22,
      USAGE         = PWR
);
MIL_USAGE = (
      MIL_STND      = 38999,
      MIL_SER       = I,
      OBJECT_ID     = PIN_1,
      OBJECT_TYPE   = TERM
);
MIL_USAGE = (
      MIL_STND      = 38999,
      MIL_SER       = II,
      OBJECT_ID     = PIN_1,
      OBJECT_TYPE   = TERM
);
TERM_CODE = (
      TERM_CODE     = SCT_1,
      PART_NUMBER   = SOCKET_1_P_N,
```

```
        TYPE            = SOCKET,
        AWG             = 20,
        WIRE_BARREL     = 22,
        USAGE           = PWR
);
MIL_USAGE = (
        MIL_STND        = 38999,
        MIL_SER         = I,
        OBJECT_ID       = SCT_1,
        OBJECT_TYPE     = TERM
);
```

Notice the one-to-many relationships on the above records. Now, we can declare the contacts and their mil usages:

```
CONTACTS = (
        CONTACT_KEY_NAME    = CK1,
        CONTACT_STYLE       = PLUG,
        INSTALLATION_TYPE   = CRIMP,

        AWG                 = 20,
        CONTACT_NAME        = A-D,

        AWG                 = 22,
        CONTACT_NAME        = 1-10
);

MIL_USAGE = (
        MIL_STND            = 38999,
        MIL_SER             = I,
        OBJECT_ID           = CK1,
        OBJECT_TYPE         = CONTACT
);
```

Notes:

1. The range of the mil_usage defined for a contact key (i.e. its mil standards and series) must be fully included within the range of the mil_usage defined for each of the termination codes.
2. The gauge (AWG) of the terminators and of the contacts must be identical.

Termination Codes

These records serve to import termination codes.

Record type - TERM_CODES
ICD Tables - TERM_CODES,
- TERM_GAUGELESS_WIRES,
- TERM_MAN_AWG,
- MATERIALS,
- ENV_CONDITIONS

Keyword	Type	Must	Remarks
TERM_CODE	string	YES	Unique
PART_NUMBER	string	YES	Unique
TYPE	string	YES	For valid types, see table below
SUBTYPE	string		For valid sub-types, see table below
VENDOR_CODE	string		Reference to the Vendor code in the VENDOR table
AWG	string		For pins, sockets and sealing plugs
WIRE_BARREL	number		For pins and sockets
WIRE_PART_NUMBER	string		Populates field GAUGELESS_WIRE_PN in table TERM_GAUGELESS_WIRES. Several Wire Part Numbers may be associated with the same terminator.
USAGE	string		From CODE_DETAIL table where code='USAGE'

Possible Values	Meaning
PWR	POWER
CX	COAX
TX	TRIAX
TW	TWINAX
FB	FIBRE OPTIC
QDRX	QUADRAX
TC-AL	Alumel
TC-CN	Constantan
TC-CR	Chromel
TC-CU	Copper
TC-FE	Iron

ENVIRON_COND	string		Environment condition is added to the table ENV_CONDITIONS, if it does not exist yet
COLOR	string		
WIRE_METAL	string		Material is added to the table MATERIALS, if it does not exist yet
MIN_CMA	number		For splices and lugs
MAX_CMA	number		For splices and lugs
MIN_N_OF_WIRES	number		For all splices: number of wires on the first side
MAX_N_OF_WIRES	number		For BUTT splices only: number of wires on the second side
STUD	string		For lugs

Keyword	Type	Must	Remarks
MIN_ WIRE_AWG	string		For lugs, pins or socket types. Per convention in the importer, for pin or socket types, this field is interpreted as the minimum wire gauge specific to this terminator (in other words, this terminator does not accommodate wires according to the table WIRE_ACCOMMODATION).
MAX_ WIRE_AWG	string		For lugs, pins or socket types. Per convention in the importer, for pin or socket types, this field is interpreted as the maximum wire gauge specific to this terminator (in other words, this terminator does not accommodate wires according to the table WIRE_ACCOMMODATION)
MIN_ WIRE_OUT_D	number		For caps and solder sleeves
MAX_ WIRE_OUT_D	number		For caps and solder sleeves
NOTE	string		
DESCRIPTION	string		
LEAD_AWG	string		For solder sleeves with lead
WEIGHT	number		

The combinations of valid types and subtypes for terminators are presented here:

Type	Sub-Type	Description
PIN		Pin
SOCKET		Socket
LUG	SINGLE_W	Lug, designed to accept a single wire
LUG	MULTI_W	Lug, designed to accept more than one wire
SPLICE	BUTT	Splice, wires come from two opposite directions
SPLICE	STUB	Splice, all wires come from same direction
CAP		Cap and Stow
SLDSLV	LEADLESS	Solder sleeve, leadless
SLDSLV	WITHBRD	Solder sleeve, with braid
SLDSLV	WITHLEAD	Solder sleeve, with lead
SOLDER		Soldering process
SEALPL		Sealing Plug, without pin
OTHER		Term code that is not one of the above listed types

Figure 4 Valid types and sub-types of terminators

Sample

TERM_CODES = (
TERM_CODE = <value>,
PART_NUMBER = <value>,
TYPE = <value>,
SUBTYPE = <value>,
VENDOR_CODE = <value>,
AWG = <value>,
WIRE_BARREL = <value>,

```
        WIRE_PART_NUMBER        = <value>,
        WIRE_PART_NUMBER        = <value>,
        WIRE_PART_NUMBER        = <value>,
        USAGE                   = <value>,
        ENVIRON_COND            = <value>,
        COLOR                   = <value>,
        WIRE_METAL              = <value>,
        MIN_CMA                 = <value>,
        MAX_CMA                 = <value>,
        MIN_N_OF_WIRES          = <value>,
        MAX_N_OF_WIRES          = <value>,
        STUD                    = <value>,
        MIN_WIRE_AWG            = <value>,
        MAX_WIRE_AWG            = <value>,
        MIN_WIRE_OUT_D          = <value>,
        MAX_WIRE_OUT_D          = <value>,
        NOTE                    = <value>,
        DESCRIPTION             = <value>,
        LEAD_AWG                = <value>,
        WEIGHT                  = <value>
);
```

Alternate Part Numbers for Termination Codes

These records serve to import alternate part numbers for terminators.

Record type - ALTERN_TERM_PN
ICD Table - ALTERN_TERM_PART_NUM

Keyword	Type	Must	Remarks
PART_NUMBER	string	YES	The terminator P/N for which an alternate is declared here
ALTERN_PART_NUMBER	string	YES	The alternate P/N for the above terminator P/N
VENDOR_CODE	string		Reference to the Vendor code in the VENDOR table

Sample

```
ALTERN_TERM_PN = (
        PART_NUMBER             = <value>,
        ALTERN_PART_NUMBER      = <value>,
        VENDOR_CODE             = <value>
);
```

Combined Termination Codes

These records serve to import components of combined terminators.

Record type - COMBO_TERM_PN
ICD Table - COMBINED_TERM_CODE_COMP

Keyword	Type	Must	Remarks	
PART_NUMBER	string	YES	The combined terminator P/N that is made of several terminator components	
COMPONENT_TERM_P_N	string	YES	P/N of the component	
SEQ_LINE_NUMBER	number		The order in which the components will be displayed for the combined terminator	
LOCATION_IN_COAX_TRIAX	string		**Possible values**	**Meaning**
			INNER	CORE
			INOUT	INNER BRAID
			OUTER	OUTER BRAID
			SHIELD	CORE/INNER BRAID

Sample

COMBO_TERM_PN = (
PART_NUMBER = <value>,
COMPONENT_TERM_P_N = <value>,
SEQ_LINE_NUMBER = <value>,
LOCATION_IN_COAX_TRIAX = <value>
);

MIL Usage

These records serve to define associations between connector standards and part specifications / terminators / contact keys.

Record type - MIL_USAGE
ICD Table - MIL_USAGE

Keyword	Type	Must	Remarks
OBJECT_ID	string	YES	Value of existing CONTACT_KEY_NAME in table CONT_KEY_GROUPS or Value of existing PART_NUMBER of a Terminator in table TERM_CODES or Value of existing PART_NUMBER for Part Specification in table PART_SPEC
OBJECT_TYPE	string	YES	"CONTACT" or "TERM" or "SPEC"
MIL_STND	string	YES	
MIL_SER	string		

Sample

MIL_USAGE = (
OBJECT_ID = <value>,
OBJECT_TYPE = <value>,
MIL_STND = <value>,
MIL_SER = <value>
);

Box Interfaces

These records serve to define box interfaces, i.e. those interface elements to which external parts and wires can be connected. There is one record per interface element (connector, stud, terminal board or pig tail) of the box specification.

Record type - BOX_INTERFACES
ICD Table - BOX_INTERFACES

Keyword	Type	Must	Remarks	
BOX_PART_NUMBER	string	YES	Box part number	
CONNECTOR_NAME	string	YES	Generic name for interface element (not designator from part occurrence) e.g. J1 or J2	
RECEPT_CONN_PART_NUMBER	string		Box Interface element P/N	
MATING_CONN_PART_NUMBER	string		Mating P/N	
CONTACT_TYPE	string		**Possible values**	**Meaning**
			CONNECTOR	Connector
			STUD	Stud
			TB	TB
			PIGTAIL	Wire-pigtail
MATING	string		Generic name for mating identification (not designator from part occurrence), e.g. P1 or P2	

Sample

BOX_INTERFACES = (
BOX_PART_NUMBER = <value>,
CONNECTOR_NAME = <value>,
RECEPT_CONN_PART_NUMBER = <value>,
MATING_CONN_PART_NUMBER = <value>,
CONTACT_TYPE = <value>,
MATING = <value>
);

Box Signals

These records serve to define signals of an interface element of a box specification.

Record type – BOX_SIGNAL
ICD Table - BOX_SIGNALS

Keyword	Type	Must	Remarks	
BOX_PART_NUMBER	string	YES	Reference to the PART_NUMBER of the box in the table PART_SPEC	
CONNECTOR_NAME	string		Reference to the CONNECTOR_NAME of the table BOX_INTERFACES	
CONTACT_NAME	string	YES	Should match CONTACT_KEY	
CONTACT_SEQ_NUMBER	number		Sequence in which contact names are displayed	
SIGNAL_NAME	string			
DIRECTION	number		**Possible values**	**Meaning**
			1	OUT
			2	IN
			3	OUT/IN
			4	FROM
			5	TO
			6	FROM/TO
			7	XMT
			8	RCV
			9	TR
EMI	string		From EMI_RULES table	
GAUGE	string		Should match CONTACT_KEY	
USAGE	string		Should match CONTACT_KEY	
SPEC_STATUS	string		"SPARE" (if the contact is spare) or left null	
LONG_SIGNAL_NAME				
SIGNAL_TYPE_ANALOG_DIG			"ANALOG" or "DIGITAL"	
VOLTAGE_NOM	number			
VOLTAGE_MIN	number			
VOLTAGE_MAX	number			
VOLTAGE_UNITS	string		"V" or "mV"	
CURRENT_NOM	number			
CURRENT_MIN	number			
CURRENT_MAX	number			
CURRENT_UNITS	string		"A" or "mA"	
FREQUENCY	number			
FREQUENCY_MIN	number			
FREQUENCY_MAX	number			
FREQUENCY_UNITS	string		"Hz" or "kHz" or "MHz" or "GHz"	
IMPEDANCE	number			
IMPEDANCE_UNITS	string		"OHM"	
NOTE	string			

Sample

BOX_SIGNAL = (

BOX_PART_NUMBER	**= <value>,**
CONNECTOR_NAME	= <value>,
CONTACT_NAME	**= <value>,**
CONTACT_SEQ_NUMBER	= <value>,
SIGNAL_NAME	= <value>,
DIRECTION	= <value>,
EMI	= <value>,
GAUGE	= <value>,
USAGE	= <value>,
SPEC_STATUS	= <value>,
LONG_SIGNAL_NAME	= <value>,
SIGNAL_TYPE_ANALOG_DIG	= <value>,
VOLTAGE_NOM	= <value>,
VOLTAGE_MIN	= <value>,
VOLTAGE_MAX	= <value>,
VOLTAGE_UNITS	= <value>,
CURRENT_NOM	= <value>,
CURRENT_MIN	= <value>,
CURRENT_MAX	= <value>,
CURRENT_UNITS	= <value>,
FREQUENCY	= <value>,
FREQUENCY_UNITS	= <value>,
IMPEDANCE	= <value>,
NOTE	= <value>,
FREQUENCY_MIN	= <value>,
FREQUENCY_MAX	= <value>,
IMPEDANCE_UNITS	= <value>

);

Wire Libraries

The following tables support the libraries of wires:

1. VENDOR (see in the part libraries section)
2. WIRE_SPEC
3. CABLE
4. WIRE_SPEC_COLOR

Wire Specification

These records serve to define specifications of wires.

Record type - WIRE_SPEC
ICD Table - WIRE_SPEC

The detail of the keywords is in next page.

Keyword	Type	Must	Remarks
PART_NUMBER	string	YES	
DIAMETER	number		Outer diameter of the whole wire/cable in length units, including the jacket
SPEC_WEIGHT	number		
SPEC_RESIST	number		See note 1.
CIRCLE	string		
WIRE_DESC	string		
VENDOR_CODE	string		Reference to the Vendor code in the VENDOR table
STRANDS	number		See note 1.
AWG	number		See note 1.
CMA	number		If missing, will be populated from GAUGE_CMA table. See note 1.
MATERIAL	string		See note 1.
INSULATION	string		See note 1.
WIRE_FORM	string		"ROUND" or "RIBBON"
CHAR_CODE	string		See note 1.
HOOK_UP	string		See note 1.
SLEEV_COLOR	number		From COLOR table: 0 - 'BLK', 1 - 'BRN', 2 - 'RED', 3 - 'ORG', 4 - 'YEL', 5 - 'GRN', 7 - 'VLT', 8 - 'GRY', 9 - 'WHT'
KINGSLEY_MARKS	number		
KINGSLEY_SLEEVE_CODE	string		From MARKING_SYSTEM table, where system = 'KINGSLEY'
TMS_SLEEVE_CODE	string		From MARKING_SYSTEM table, where system = 'TMS'
MARKABLE_WIRE	string		Y or N
USAGE	string		From CODE_DETAIL table where code ='USAGE' and value not like 'TC%' or 'TC' ; See also note 1.
WIRE_STYLE	string	YES	See details in the table below
SHIELD_LAYERS	number		
CONDUCTORS_NUM	number		Equal to number of "basic colors"
OUTER_INSULATION	string		
H_CABLE	string		
WIRE_DIAMETER	number		for standard cables, this value is the outer diameter of each of the individual wires that are part of the cable, expressed in length units. See note 1.
IMPEDANCE	number		from r16a.00 version, in Ohm for triax and coax wires; See note 1.
BASIC_WIRE_PART_NUMBER	string		When WIRE_STYLE=STANDARD (i.e. for standard cables), this field refers to the Part_number of an existing wire specification whose WIRE_STYLE = STANDARD_WIRE. T
PLATING	string		See note 1.

Here are the various values for the mandatory field WIRE_STYLE:

Value	Name (GUI)	Description
"STANDARD"	Standard Cable	for all single wires and standard cables (e.g. twisted, coax, shielded or triax**)**
"CABLE"	Hybrid Cable	for hybrid cables
"IN_HYBRID_ONLY"	Use in hybrid cable only	for single wires and cables that can be used as components of hybrid cable only
"STANDARD_WIRE"	Wire	for single wires (r16a.00 version)

Figure 5 Values for field WIRE_STYLE

Note 1: *In the case of a standard cable referring to the specification of its basic wires, it is recommended not to provide any value for this field. The importer program will copy the proper value from the basic wire specification to the empty field.*
In the case where a value is nevertheless provided for this field, then it is mandatory that the provided value be equal to the value of the correspondent field of the basic wire specification.

Sample

WIRE_SPEC(
PART_NUMBER = <value>,
DIAMETER = <value>,
SPEC_WEIGHT = <value>,
SPEC_RESIST = <value>,
CIRCLE = <value>,
WIRE_DESC = <value>,
VENDOR_CODE = <value>,
STRANDS = <value>,
AWG = <value>,
CMA = <value>,
MATERIAL = <value>,
INSULATION = <value>,
WIRE_FORM = <value>,
CHAR_CODE = <value>,
HOOK_UP = <value>,
SLEEV_COLOR = <value>,
KINGSLEY_MARKS = <value>,
TMS_MARKS = <value>,
KINGSLEY_SLEEVE_CODE = <value>,
TMS_SLEEVE_CODE = <value>,
MARKABLE_WIRE = <value>,
USAGE = <value>,
WIRE_STYLE = <value>,
SHIELD_LAYERS = <value>,
CONDUCTORS_NUM = <value>,
PLATING = <value>,
OUTER_INSULATION = <value>,
H_CABLE = <value>,
WIRE_DIAMETER = <value>,
IMPEDANCE = <value>,
BASIC_WIRE_PART_NUMBER = <value>
);

Wire Spec Color

These records serve to import colors and usages of basic wires of a cable.

Record type - WIRE_SPEC_COLOR
ICD Table - WIRE_SPEC_COLOR

Keyword	Type	Must	Remarks
PART_NUMBER	string	YES	WIRE_SPEC.PART_NUMBER = WIRE_SPEC_COLOR.PART_NUMBER
COLOR	string	YES	
USAGE	string		From CODE_DETAIL table, where CODE = 'USAGE' and VALUE like 'TC%'

Sample

WIRE_SPEC_COLOR = (
PART_NUMBER = <value>,
COLOR = <value>,
USAGE = <value>
);

CABLE

These records serve to import the components of a hybrid cable.

Record type - CABLE
ICD Table - CABLE

Keyword	Type	Must	Remarks
CABLE_NUMBER	string	YES	WIRE_SPEC.PART_NUMBER = CABLE.CABLE_NUMBER
PART_NUMBER	string	YES	
QTY	number		
WIRE-IDENT	string		
SEQUENCE	number		

Sample

CABLE = (
CABLE_NUMBER = <value>,
PART_NUMBER = <value>,
QTY = <value>,
WIRE_IDENT = <value>,
SEQUENCE = <value>
);

User Interface

Invocation

The import program is initiated from the "harness ansi" menu.
At the operating system prompt, **enter**:

harness ansi

Figure 6: Invocation of* "harness *ansi" operations

Then, the system displays the following menu:

```
Ansi terminal Options:
======================
  1 ) file_convert
  2 ) csr_convert
  3 ) importer
  4 ) drawing_generator
  5 ) flag_legend_check
  6 ) Exit ( or 'q' )

Select option number:
```

Figure 7:* "harness *ansi" Options

By selecting option 3, the ***HarnesSys*** Importer function is chosen.

Importer Main Menu

The Importer main menu looks like this:

```
Importer to HarnesSys
-------------------------
  1 ) Import one single transaction file at a time
  2 ) Define a scenario (sequence of import operations)
  3 ) Run a scenario
  Q ) Quit

Select option >>>
```

Figure 8:* HarnesSys *Importer Main Menu

Option 1: Import one single transaction file at a time

This option imports transaction (import) files, one by one. For most cases, the user interaction is presented below. Variations are described afterwards.

Menu prompts	Explanation
Enter Project (3 letters) >>>	The aircraft database project. Entry is not case-sensitive.
Actions : ------- 1) Import Documents 2) Import Part Occurrences 3) Tabular Schematics operations 4) Import Free Pins 5) Import Manufacturing Data 6) Import ETA Test Bundle 7) Import Circuit Analyzer 8) Import Retrofit Condition Notes 9) Import Box Signals/Part/Wire Spec Libraries Select option [<CR> exit] >>>	Choose a number 9. Choose <CR> to cancel the pending operation and return to the Importer main menu. The menu dialog is as below.
Enter Import File Name [<CR> to exit] >>>	The import file name (accept relative or full directory path) <CR>: back to the main menu. The menu dialog is as below.
Do you want to overwrite the existing data y/n [n]?	N – means to add the new values to the existing library, Y – Means to delete the existing values in the library and import the values from the Import file.
---------- Import Options are: ---------- Project .....……*ABC* Import task on .....PART_WIRE_SPEC_DATA Transaction file .....*YYY* Document name ..... From line .....…...beginning of file To line .....….....end of file Debug assistance…..OFF Action: Import Transaction file O.K.? y/n [y]	The user keyboard entries (here *ABC*) are summarized for confirmation. Enter 'y' to confirm the options & launch the import. Enter 'n' to step back to the importer main menu. Default is 'y'. Next lines will appear if you entered Y.
Import Part/Wire Specification Libraries ……. …… ……..	You will get messages about the inserted data.

Please check errors/messages file ~<*user*>/harness/<*project*>/private/Import_Lib.msg Do you want to commit the changes to the database? Enter 'C' to COMMIT or 'S' to STOP:	
Next import? Actions : ------- 1) Import Documents 2) Import Part Occurrences 3) Tabular Schematics operations 4) Import Free Pins 5) Import Manufacturing Data 6) Import ETA Test Bundle 7) Import Circuit Analyzer 8) Import Retrofit Condition Notes 9) Import Box Signals/Part/Wire Spec Libraries Select option [<CR> exit] >>>	After the import, it is possible to proceed to next import task for the same project. <CR>: back to the main menu.

Figure 9: Single Import Dialog

Note: The library import operations are not supported in the Scenario mode.

Tracking the Importer Actions

In order to review what happened during the execution of a given library import operation, HarnesSYS generates several files that allow the user to track the effect of the import. These files are all saved in the private directory of the user. They include:

1. A log file (named *Import_Lib.log*), that holds the history of the interaction during the session,
2. A 'message' file (named *Import_Lib.msg*), that summarizes the encountered warnings and errors, and the quantity of proceeded elements,
3. A SQL script file, (named *import_results_YYYYmmDDHHMMSS.sql*). This time-stamped script can be run just after changes are committed. It will query the database about the entities that were imported or updated. The exact way to invoke this script is detailed in the last lines of the .msg file.

   The result of the query is saved in a file (named *import_results_YYYYmmDDHHMMSS.lst)*, with the same time stamp, and also located in the private directory.