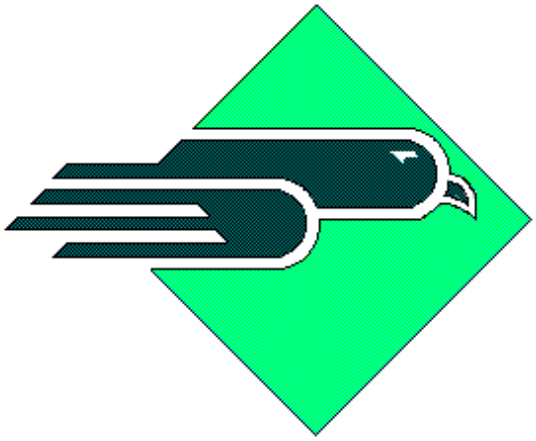

WESTERN AVIONICS

3838 / 3910 Remote Terminal Validation Test Plan
Software Driver Library

P/N 1L01649H01 Rev 1.2

User Manual
UM 01649 Rev C

13/14 Shannon Free Zone
Co. Clare
Ireland

17 December 1996

TABLE OF CONTENTS

1. GENERAL INFORMATION

1.1. Introduction

The VPLAN software package is a library of 'C' functions designed to permit easy implementation of both the 3838 and 3910 RT Validation Test Plan using a Western Avionics 4210, 2110 VME or 4510 PC 1553/3910 interface card. This library is supplied as source code; to be compiled and linked with a user defined application.

The files can be split into three categories:

1. LS 3838 Validation files.
These files all begin with 'vp' followed by a four digit number. This 4 digit number corresponds to a section of the RT Validation Test Plan. For example, vp5217.c carries out the tests outlined in 5.2.1.7 of the RT Validation Test Plan.

2. HS 3910 Validation files.
These files all begin with 'hp' followed by a three digit number. This 3 digit number corresponds to a section of the prEN 3910 RT Validation Test Plan. Each file will carry out the tests as close as possible as outlined in prEN 3910 RT Validation Plan. For example, hp545.c carries out the tests outlined in 5.4.5 of the prEN 3910 RT Validation Test Plan.

3. General Files.
The file vprdwr.c MUST be compiled and linked with any program, HS or LS. The files vplan.h and vpflags.h MUST reside in the application directory for compilation.

2. INSTALLATION AND PREPARATION FOR USE

2.1. General

The library is supplied on floppy disk in IBM PC format. The library is supplied as source code in 'C'. It is a collection of files, all of which must be copied to a suitable working directory. The user should refer to the compiler vendors manuals which describe the requirements for the user specific operating system. Refer also to the user manual for the particular card being used. The following files are contained on the disk:-

vplan.h	vp5211.c	vp5221.c	hp541.c	hp555.c
vpflags.h	vp5212.c	vp5222.c	hp542.c	hp556.c
	vp5213.c	vp5223.c	hp543.c	hp561.c
vprdwr.c	vp5214.c	vp5224.c	hp544.c	hp562.c
	vp5215.c	vp5225.c	hp545.c	
	vp5216.c		hp551.c	
	vp5217.c		hp552.c	
	vp5218.c		hp553.c	
	vp5219.c		hp554.c	

2.2. Compiler Variations

The library has been designed to be used for a wide range of 'C' compilers.

To allow easy portability a convention must be followed by the programmer using the library.

All VPLAN functions return a data type 'Error'. The value of this 'Error' is not related to the success or failure of the particular validation test, it is set according to the validity of the input parameters range, syntax, etc. For good input parameters the function should always return 'Error' equal to NO_ERROR as defined in VPLAN.H.

All VPLAN functions require a pointer to a structure of type vpHandle. This structure contains all information about the card, including type and address information.

All function input parameters and results are of the following types:

Ubyte (unsigned 8 bits)
Sbyte (signed 8 bits)
Uword (unsigned 16 bits)
Sword (signed 16 bits)
Ulong (unsigned 32 bits)
Ulong (signed 32 bits)
Error (unsigned 16 bits)

Note:

- All application programs using the library MUST use these types to ensure data size compatibility. The definition of these types is done by selecting the appropriate value for PC_COMP in vpflags.h.
- As the size of 'NULL' can vary from compiler to compiler and hence cause confusion when porting from system to system the use of this has been avoided. Instead, the value 'NO_PTR' is used to define a NULL pointer.

2.3. vpflags.h

The header file vpflags.h will need to be edited to suit the particular compiler and card type being used. The beginning of vpflags.h has a list of five possible environments that the drivers can be compiled for:

1. PLATFORM_VME_CARD 2110/4210 VME card
2. PLATFORM_DOS_PCCARD 4510 PC card under DOS
3. PLATFORM_WIN_PCCARD 4510 PC card under WINDOWS
4. PLATFORM_DOS_PASSERELLE PASSERELLE PC/VME interface (DOS)
5. PLATFORM_WIN_PASSERELLE PASSERELLE PC/VME interface (WINDOWS)

The desired environment must be defined as '1'. All others must be defined as '0'.

The definitions following this list in vpflags.h are sub-definitions for the particular environment as follows:

- __STDC__ If defined as '1' the code will be compiled as ANSI C. If defined as '0' the code will be compiled to the Kernigan and Ritchie standard.

- PC_COMP All code in the library uses a set of data types as follows:

TYPE	PC_COMP = 0	PC_COMP = 1
Ubyte (unsigned 8 bits)	unsigned char	unsigned char
Sbyte (signed 8 bits)	signed char	signed char
Uword (unsigned 16 bits)	unsigned short	unsigned int
Sword (signed 16 bits)	signed short	signed int
Ulong (unsigned 32 bits)	unsigned int	unsigned long
Slong (signed 32 bits)	signed int	signed long
Error (unsigned 16 bits)	unsigned short	unsigned int

3. INITIALISATION FUNCTIONS

3.1. vpInit()

Function:-

Error vpInit(vpHandle *cardHandle, Ulong base);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
base	This is the VME base address of the card to be used.

Description:-

The vpInit() function is used to prepare the vpHandle of a VME card for use by the other VPLAN functions. The pointer to this structure is used as a 'handle' by the other driver functions to uniquely identify the board. If the target hardware is purely VME then this function is all that is required to initialise the cardHandle. The function vpInitPC() is not used.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle        card1;
Error           error;
...
error = vpInit(&card1, 0xC00000);
```

3.2. vpInitPC()

Function:-

Error vpInitPC(vpHandle *cardHandle, vpPCinfo *info);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vpPCinfo structure defining PC initialisation options.

Description:-

The vpInitPC() function is used to prepare the vpHandle of a PC card for use by the other VPLAN functions. The pointer to this structure is used as a 'handle' by the other driver functions to uniquely identify the board. If the target hardware is a PC 4510 or a VME/PC gateway card (Passerelle) then this function is all that is required to initialise the cardHandle. The function vpInit() is not used.

The vpPCinfo structure is:

```
typedef struct
{
   Uword portBaseAdrs;
   Ulong  cardAddress;
   Ulong  pcAddress;
   Uword  segSize;
   Uword  lsCoupling;
   Uword  memorySize;
 }vpPCinfo;
```

The elements of the vpPCinfo structure should be set as follows :-

portBaseAdrs:
Base address for PC card I/O registers.

cardAddress:
VME base address (only if using VME/PC gateway card).

pcAddress:
Base address in PC for memory segment.

segSize:
Segment size in Kbytes(8,16,32,64 or 128).

lsCoupling:
3838 coupling option (DIRECT_COUPLING or STUB_COUPLING).

memorySize:
Data width for PC access (PC_MEM_SIZE_08 or PC_MEM_SIZE_16).

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle          card1;
vpPCinfo          info;
Error             error;
...
info.portBaseAdrs  = 0x300;
info.cardAddress    = 0xC00000;
info.pcAddress      = 0xE0000;
info.segSize         = 8;
info.lsCoupling      = DIRECT_COUPLING;
info.memorySize     = PC_MEM_SIZE_16;

error = vpInit(&card1, &info);
```

4. LS FUNCTIONS

4.1. vp5211()

Function:-

Error vp5211(vpHandle *cardHandle, vp5211Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp5211Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 5211 test failed for any reason.

Description:-

Function vp5211() carries out the test requirements of paragraph 5.2.1.1 of the RT Validation Test Plan on the UUT. The values in the vp5211Info structure are for specifying various options and variables and reporting back detailed result information.

The vp5211Info structure is:

```
typedef struct
{
   Uword  rtNo;
   Ulong  errCountC;
   Ulong  errCountR;
   Ulong  errCountT;
   Uword  *errListC;
   Uword  *errListR;
   Uword  *errListT;
   Ulong  listSizeC;
   Ulong  listSizeR;
   Ulong  listSizeT;
   Uword  step1Cmd;
   Uword  rtrtRxCmd;
   Uword  rtrtTxCmd;
   Ubyte  modeDefs[32];
   Ubyte  subDefs[32];
   Uword  *txData;
   Ubyte  txLast;
   Ubyte  bCasts;
   Ubyte  illegal;
}vp5211Info;
```

The elements of the vp5211Info structure should be set as follows :-

portBaseAdrs:
Base address for PC card I/O registers.

rtNo:
Address of UUT (0 to 30)

errCount:
This is set by the function to report the total number of messages that failed the 5211 test.

errCountR:
This is set by the function to report the total number of RX messages that failed the 5211 test.

errCountT:
This is set by the function to report the total number of TX messages that failed the 5211test.

***errListC:**
This is a pointer to an array of Uwords for storing a list of all the commands that failed the 5211 test.

***errListR:**
This is a pointer to an array of Uwords for storing a list of all the RX commands that failed the 5211 test.

***errListT:**
This is a pointer to an array of Uwords for storing a list of all the TX commands that failed the 5211 test.

listSizeC:
This value shall be set by the user to define the maximum length of the errListC array.

listSizeR:
This value shall be set by the user to define the maximum length of the errListR array.

listSizeT:
This value shall be set by the user to define the maximum length of the errListT array.

step1Cmd:
This shall be set to define the desired command to be used for step 1 of the 5211 test.

rtrtRxCmd:
This shall be set to define the RX command to the UUT for the RT-RTmessage section of 5211.

rtrtTxCmd:
This shall be set to define the TX command to the UUT for the RT-RT message section of 5211.

modeDefs[32]:
This 32 Ubyte array shall define the mode codes implemented by the UUT. If the element of the array is TRUE then the UUT will be tested for that mode code.

subDefs[32]:
This 32 Ubyte array shall define the maximum allowable words for each subaddress. A value of zero will define the subaddress as not implemented.

***txData:**
This 32 Uword array will define any specific TX data to be used in the test. If this value is set to 'NO_PTR' then the function will use its own default data values.

txLast:
This flag is set TRUE by the user if the UUT is capable of responding to the TX LAST Command mode code.

bCasts: This flag is set TRUE by the user if the UUT is capable of implementing broadcast commands.

illegal:
This flag is set TRUE by the user if the UUT implements the illegal message option.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle        card1;
vp5211Info      info;
Uword           i, failed;
Uword           listC[10], listR[10], listT[10];
Error           error;
...

info.rtNo       = 1;
info.errListC   = listC;
info.errListR   = listR;
info.errListT   = listT;
info.listSizeC  = 10;
info.listSizeR  = 10;
info.listSizeT  = 10;
info.step1Cmd   = 0x0821;
info.rtrtRxCmd  = 0x0820;
info.rtrtTxCmd  = 0x0C20;
info.txData     = NO_PTR;
info.txLast     = 1;
info.bCasts     = 1;
info.illegal    = 0;

for(i=0; i!=32; i++)
{
   info.modeDefs[i] = 1;
   info.subDefs[i]   = 1;
}

error = vp5211(&card1, &info, &failed);
```

4.2. vp5212()

Function:-

Error vp5212(vpHandle *cardHandle, vp5212Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp5212Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 5212 test failed for any reason.

Description:-

Function vp5212() carries out the test requirements of paragraph 5.2.1.2 of the RT Validation Test Plan on the UUT. The values in the vp5212Info structure are for specifying various options and variables and reporting back detailed result information.

The vp5212Info structure is:

```
typedef struct
{
   Uword  stage;
   Uword  msg1Cmnd1;
   Uword  msg1Cmnd2;
   Uword  msg1RTRT;
   Uword  msg1Bcst;
   Uword  msg1Sim1;
   Uword  msg1Sim2;
   Uword  *txData1;
   Uword  msg2Cmnd1;
   Uword  msg2Cmnd2;
   Uword  msg2RTRT;
   Uword  msg2Bcst;
   Uword  msg2Sim1;
   Uword  msg2Sim2;
   Uword  *txData2;
}vp5212Info;
```

The elements of the vp5212Info structure should be set as follows :-

stage:
Part of 5212 test to be carried out (0=52121, 1=52122).

msg1Cmnd1:
Command word for 1st message.

msg1Cmnd2:
RT-RT command word for 1st message.

msg1RTRT:
1st message RT-RT message type flag.

msg1Bcst:
1st message Broadcast type flag.

msg1Sim1:
1st message Simulate 1st RT response.

msg1Sim2:
1st message Simulate 2nd RT response.

***txData1:**
1st message TX data (can be set to NO_PTR for default data).

msg2Cmnd1:
Command word for 2nd message.

msg2Cmnd2:
RT-RT command word for 2nd message.

msg2RTRT:
2nd message RT-RT message type flag.

msg2Bcst:
2nd message Broadcast type flag.

msg2Sim1:
2nd message Simulate 1st RT response.

msg2Sim2:
2nd message Simulate 1st RT response.

***txData2:**
2nd message TX data (can be set to NO_PTR for default data).

Note:

- The vp5212() function allows any 2 user defined messages to be transmitted with minimum intermessage gap times. The user must initialise the elements of the structure to indicate if the message type is standard, RT-RT, broadcast etc. If the message type is RT-RT then the simulation flags have to be set to indicate which part of the RT-RT transfer is carried out by the UUT and which part is carried out by the test equipment.
- The 'stage' flag if clear will result in the message pair being sent in excess of 1000 times as required by 52121 of the validation plan. If any of the messages fail then the 'failed' flag will be returned set TRUE. The 'stage' flag if set will result in the message pair being sent for a duration of greater than 30 seconds as required by 52122 of the validation plan. If any of the messages fail then the 'failed' flag will be returned set TRUE.
- It is the responsibility of the applications programmer to choose suitable message pairs and repetitively call vp5212() to meet all the permutations defined in paragraph 5212 of the RT Validation Test Plan.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle        card1;
vp5212Info      info;
Uword           failed;
Error           error;
...

info.stage      = 1;
info.msg1Cmnd1 = 0x082A;
info.msg1Cmnd2 = 0x143A;
info.msg1RTRT  = 1;
info.msg1Bcst   = 0;
info.msg1Sim1   = 1;
info.msg1Sim2   = 0;
info.txData1    = NO_PTR;
info.msg2Cmnd1 = 0x0C2A;
info.msg2Cmnd2 = 0;
info.msg2RTRT  = 0;
info.msg2Bcst   = 0;
info.msg2Sim1   = 0;
info.msg2Sim2   = 0;
info.txData2    = NO_PTR;

error = vp5212(&card1, &info, &failed);
```

4.3. vp5213()

Function:-

Error vp5213(vpHandle *cardHandle, vp5213Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp5213Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 5213 test failed for any reason.

Description:-

Function vp5213() carries out the test requirements of paragraph 5.2.1.3 of the RT Validation Test Plan on the UUT. The values in the vp5213Info structure are for specifying various options and variables and reporting back detailed result information.

The vp5213Info structure is:

```
typedef struct
{
   Uword  rtNo;
   Uword  step1Cmd;
   Uword  txCommand;
   Uword  rxCommand;
   Uword  rxMode;
   Uword  *txData;
   Ulong  failSafeTout;
   Uword  failSafeStep;
   Uword  results[VP_TOT_5213];
}vp5213Info;
```

The elements of the vp5213Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30)

step1Cmd:
This shall be set to define the desired command to be used for step 1 of the 5213 test.

txCommand:
This shall be set to define the desired TX command to be used for the 5213 test.

rxCommand:
This shall be set to define the desired RX command to be used for the 5213 test.

rxMode:
This shall be set to define the desired RX mode code to be used for the 5213 test.

***txData:**
TX data for test equipment (can be set to NO_PTR for default data).

failSafeTout:
Desired failsafe timeout.

failSafeStep:
Failsafe stage for test if any.

results[]:
Results of 5213 test.

Note:
- If failSafeStep is set to VP_NO_TEST then vp5213() will carry out all the error injection tests as defined by 5213 of the RT Validation Test Plan.
- If failSafeStep is set to VP_FAILSAFE_STEP_1 the test equipment will wait failSafeTout while the user initiates the condition causing the fail-safe timer to timeout. The test equipment will measure the duration of activity.
- If failSafeStep is set to VP_FAILSAFE_STEP_3 the test equipment will send the command as defined in step 3 and report any error.

results[] shall contain an itemised failure report as follows:-

results[VP_PARITY_TEST]	set if PARITY error test failed.
results[VP_WORDLENGTH_TEST]	set if WORDLENGTH error test failed.
results[VP_MANCHESTER_TEST]	set if MANCHESTER error test failed.
results[VP_SYNC_TEST]	set if SYNC error test failed.
results[VP_MSGLENGTH_TEST]	set if MSGLENGTH error test failed.
results[VP_CONTIGUOUS_TEST]	set if CONTIGUOUS error test failed.
results[VP_FAILSAFE_TEST]	set if FAILSAFE error test failed.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle        card1;
vp5213Info      info;
Uword           failed;
Error           error;
...

info.rtNo          = 1;
info.step1Cmd      = 0x0821;
info.txCommand     = 0x0C2A;
info.rxCommand     = 0x082A;
info.rxMode        = 0x0811;
info.failSafeTout  = 60;
info.failSafeStep  = VP_NO_TEST;

error = vp5213(&card1, &info, &failed);
```

4.4. vp5214()

Function:-

Error vp5214(vpHandle *cardHandle, vp5214Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp5214Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 5214 test failed for any reason.

Description:-

Function vp5214() carries out the test requirements of paragraph 5.2.1.4 of the RT Validation Test Plan on the UUT. The values in the vp5214Info structure are for specifying various options and variables and reporting back detailed result information.

The vp5214Info structure is:

```
typedef struct
{
   Uword  rtNo;
   Uword  txCommand;
   Uword  rxCommand;
   Uword  rtrtRxCmd;
   Uword  *txData;
   Uword  results[VP_TOT_5214];
}vp5214Info;
```

The elements of the vp5214Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30)

txCommand:
This shall be set to define the desired TX command to be used for the 5214 test.

rxCommand:
This shall be set to define the desired RX command to be used for the 5214 test.

rtrtRxCmd:
This shall be set to define the desired RX command for the RT-RT section of the 5214 test.

***txData:**
TX data for test equipment (can be set to NO_PTR for default data).

results[]:
Results of 5214 test.

results[] shall contain an itemised failure report as follows:-

results[VP_STD_SUPERSEDING_TEST] set if failed standard superseding test.
results[VP_RTRT_SUPERSEDING_TEST] set if failed RT-RT superseding test.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle        card1;
vp5214Info      info;
Uword           failed;
Error           error;
...

info.rtNo       = 1;
info.txCommand  = 0x0C2A;
info.rxCommand  = 0x082A;
info.rtrtRxCmd  = 0x082A;
info.txData     = NO_PTR;

error = vp5214(&card1, &info, &failed);
```

4.5. vp5215()

Function:-

Error vp5215(vpHandle *cardHandle, vp5215Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp5215Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 5215 test failed for any reason.

Description:-

Function vp5215() carries out the test requirements of paragraph 5.2.1.5 of the RT Validation Test Plan on the UUT. The values in the vp5215Info structure are for specifying various options and variables and reporting back detailed result information.

The vp5215Info structure is:

```
typedef struct
{
   Uword  rtNo;
   Uword  rxCommand;
   Uword  modeXXXXX;
   Uword  results[VP_TOT_5215];
}vp5215Info;
```

The elements of the vp5215Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30)

rxCommand:
This shall be set to define the desired RX command to be used for the 5215 test.

modeXXXXX:
Mode code bits value 00000 or 11111.

results[]:
Results of 5215 test.

results[] shall contain an itemised failure report as follows:-

results[VP_TSTATUS_TEST]	set if failed TX STATUS mode test
results[VP_SHUTDOWN_TEST]	set if failed TX SHUTDOWN mode test
results[VP_RESET_REMOTE_TEST]	set if failed RESET TERMINAL mode test

Note:
- This function should be executed twice. Once with modeXXXXX = 0x00 and once with modeXXXXX = 0x1F.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle          card1;
vp5215Info        info;
Uword             failed;
Error             error;
...

info.rtNo              = 1;
info.rxCommand     = 0x082A;
info.modeXXXXX   = 0x00;

error = vp5215(&card1, &info, &failed);

if(error == NO_ERROR)
{
   info.modeXXXXX = 0x1F;
   error = vp5215(&card1, &info, &failed);
}
```

4.6. vp5216()

Function:-

Error vp5216(vpHandle *cardHandle, vp5216Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp5216Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 5216 test failed for any reason.

Description:-

Function vp5216() carries out the test requirements of paragraph 5.2.1.6 of the RT Validation Test Plan on the UUT. The values in the vp5216Info structure are for specifying various options and variables and reporting back detailed result information.

The vp5216Info structure is:

```
typedef struct
{
   Uword  rtNo;
   Uword  subad;
   Uword  wcnt;
}vp5216Info;
```

The elements of the vp5216Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30)

subad:
RT subaddress for wrap-round data.

wcnt:
Maximum number of words permitted for wrap-round

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle        card1;
vp5216Info      info;
Uword           failed;
Error           error;
...

info.rtNo       = 1;
info.subad      = 30;
info.wcnt       = 31;

error = vp5216(&card1, &info, &failed);
```

4.7. vp5217()

Function:-

Error vp5217(vpHandle *cardHandle, vp5217Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp5217Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 5217 test failed for any reason.

Description:-

Function vp5217() carries out the test requirements of paragraph 5.2.1.7 of the RT Validation Test Plan on the UUT. The values in the vp5217Info structure are for specifying various options and variables and reporting back detailed result information.

The vp5217Info structure is:

```
typedef struct
{
   Uword  rtNo;
   Uword  subad;
   Uword  wcnt;
   Uword  tCmd;
   Uword  *txData;
   Uword  results[VP_TOT_5217];
}vp5217Info;
```

The elements of the vp5217Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30)

subad:
RT-RT UUT subaddress

wcnt:
RT-RT maximum wordcount for test

tCmd:
Used by function only

***txData:**
TX data for test equipment (can be set to NO_PTR for default data).

results[]:
Results of 5215 test.

results[] shall contain an itemised failure report as follows:-

results[VP_RTRT_TIMEOUT_TEST]	set if failed RT-RT timeout test
results[VP_RTRT_RX_TEST]	set if failed RT-RT RX data test.
results[VP_RTRT_TX_TEST]	set if failed RT-RT TX data test.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle        card1;
vp5217Info      info;
Uword           failed;
Error           error;
...

info.rtNo       = 1;
info.subad      = 8;
info.wcnt       = 31;
info.txData     = NO_PTR;

error = vp5217(&card1, &info, &failed);
```

4.8. vp5218()

Function:-

Error vp5218(vpHandle *cardHandle, vp5218Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp5218Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 5218 test failed for any reason.

Description:-

Function vp5218() carries out the test requirements of paragraph 5.2.1.8 of the RT Validation Test Plan on the UUT. The values in the vp5218Info structure are for specifying various options and variables and reporting back detailed result information.

The vp5218Info structure is:

```
typedef struct
{
   Uword  rtNo;
   Uword  mode;
   Uword  txCommand;
   Uword  delay;
   Uword  timeout;
   Uword  result;
}vp5218Info;
```

The elements of the vp5218Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30)

mode:
This shall define the type of interrupting command to be used. The value of this shall be set to one of three values:

VP_5218_STANDARD	- Use good TX command without errors.
VP_5218_PARITY	- Set parity error in TX command.
VP_5218_WRONG_ADDRESS	- Set address to different RT.

txCommand:
This shall be set to define the desired TX command to be used as the interrupting command.

delay:
This value shall define the overlap of the messages. The resolution of this is 200nS.

timeout:
Timeout in seconds that the function will wait for the external message to start.

result:
Result of 5218 test. Set if test failed.

Note:

- This function should be executed repeatedly with increasing values of delay that meet the requirements of 5218 (increase from 4uS until there is no overlap).
- The test equipment does not generate the initial message only the overriding message. It is the responsibility of the user to provide a master source of 1553 to generate the initial message.
- To implement the full test of 5218 the user must call vp5218() prior to sending the message to be interrupted. Once the routine is called, the user must then start the initiating message.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle        card1;
vp5218Info      info;
Uword           failed;
Error           error;
...

info.rtNo       = 1;
info.mode       = VP_5218_PARITY;
info.txCommand  = 0x0C2A;
info.delay      = 20;

error = vp5218(&card1, &info, &failed);

/* NOW START EXTERNAL SOURCE! */
```

4.9. vp5219()

Function:-

Error vp5219(vpHandle *cardHandle, vp5219Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp5219Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 5219 test failed for any reason.

Description:-

Function vp5219() carries out the test requirements of paragraph 5.2.1.9 of the RT Validation Test Plan on the UUT. The values in the vp5219Info structure are for specifying various options and variables and reporting back detailed result information.

The vp5219Info structure is:

```
typedef struct
{
   Uword  rtNo;
   Uword  stepNo;
   Uword  step1Cmd;
   Uword  *txData;
}vp5219Info;
```

The elements of the vp5219Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30)

stepNo:
Step number of 5219 test (2,3 or 4)

step1Cmd:
Command for using in step 1 of test.

***txData:**
TX data for test equipment (can be set to NO_PTR for default data).

Note:

- This function should be executed 3 times with values 2,3 and 4 respectively for stepNo. After each execution the user must make the relevant hardware modifications to the UUT before proceeding with the next call. At the end of the final execution of vp5219() the value of 'failed' will be 0 if the test has passed.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle        card1;
vp5219Info      info;
Uword           failed;
Error           error;
...

info.rtNo       = 1;
info.stepNo     = 2;
info.step1Cmd   = 0x082A;
info.txData     = NO_PTR;

error = vp5219(&card1, &info, &failed);
```

4.10. vp5221()

Function:-

Error vp5221(vpHandle *cardHandle, vp5221Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp5221Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 5221 test failed for any reason.

Description:-

Function vp5221() carries out the test requirements of paragraph 5.2.2.1 of the RT Validation Test Plan on the UUT. The values in the vp5221Info structure are for specifying various options and variables and reporting back detailed result information.

The vp5221Info structure is:

```
typedef struct
{
   Uword  rtNo;
   Uword  modes[VP_TOT_5221];
   Uword  rxCommand;
   Uword  txCommand;
   Uword  priID;
   Uword  secID;
   Uword  modeXXXXX;
   Uword  *txData;
   Uword  results[VP_TOT_5221];
}vp5221Info;
```

The elements of the vp5221Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30)

modes:
Array defining which mode codes require testing

txCommand:
This shall be set to define the desired TX command to be used for the 5221 test.

rxCommand:
This shall be set to define the desired RX command to be used for the 5221 test.

priID:
Word to be used by the SELECTIVE SHUTDOWN modes to define the primary bus.

secID:
Word to be used by the SELECTIVE SHUTDOWN modes to define the secondary bus.

modeXXXXX:
Mode code bit pattern to use (00000 or 11111)

***txData:**
TX data for test equipment (can be set to NO_PTR for default data).

results[]:
Results of 5221 test.

Note:

- The 'modes' array have the following elements:

VP_DBCA_TEST	- Dynamic bus control acceptance
VP_SYNZ_WO_TEST	- Synchronise without data word
VP_SYNZ_WI_TEST	- Synchronise with data word
VP_STEST_TEST	- Initiate selftest
VP_BITWD_TEST	- Transmit BIT word
VP_SELTX_TEST	- Selective transmitter shutdown
VP_TINHB_TEST	- Transmit inhibit
VP_TVECT_TEST	- Transmit vector word
VP_TLAST_TEST	- Transmit last command

  Each one of these elements must be initialised to VP_NO_TEST or VP_DO_TEST.
- The results array have the same elements as the 'modes' array. If a particular mode code is tested and fails, the corresponding results array element will be set TRUE.
- This function should be executed twice. Once with modeXXXXX = 0x00 and once with modeXXXXX = 0x1F;

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle          card1;
vp5221Info        info;
Uword             failed;
Uword             i;
Error             error;
...

info.rtNo              = 1;
info.rxCommand     = 0x082A;
info.txCommand     = 0x0C2A;
info.priID             = 0x0000;
info.secID             = 0x0001;
info.modeXXXXX    = 0x001F;
info.txData            = NO_PTR;

for(i=0; i < VP_TOT_5221; i++)
{
   info.modes[i] = VP_DO_TEST;
}

error = vp5221(&card1, &info, &failed);
```

4.11. vp5222()

Function:-

Error vp5222(vpHandle *cardHandle, vp5222Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp5222Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 5222 test failed for any reason.

Description:-

Function vp5222() carries out the test requirements of paragraph 5.2.2.2 of the RT Validation Test Plan on the UUT. The values in the vp5222Info structure are for specifying various options and variables and reporting back detailed result information.

The vp5222Info structure is:

```
typedef struct
{
   Uword  rtNo;
   Uword  sbits[VP_TOT_5222];
   Uword  rxCommand;
   Uword  txCommand;
   Uword  rxBcast;
   Uword  *txData;
   Uword  results[VP_TOT_5222];
}vp5222Info;
```

The elements of the vp5222Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30)

sbits:
Array defining which status word bits to test

rxCommand:
This shall be set to define the desired RX command to be used for the 5222 test.

txCommand:
This shall be set to define the desired TX command to be used for the 5222 test.

rxBcast:
This shall be set to define the desired RX broadcast command to be used for the 5222 test.

***txData:**
TX data for test equipment (can be set to NO_PTR for default data).

results[]:
Results of 5222 test.

Note:

- The 'sbits' array have the following elements:
  VP_SRQ_TEST - Service request bit
  VP_BCAST_TEST - Broadcast received bit
  VP_BUSY_TEST - Busy bit
  VP_SUBSYS_TEST - Subsystem flag bit
  VP_TFLAG_TEST - Terminal flag bit
  Each one of these elements must be initialised to VP_NO_TEST or VP_DO_TEST.
- The results array have the same elements as the 'sbits' array. If a particular status bite is tested and fails, the corresponding results array element will be set TRUE.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle        card1;
vp5222Info      info;
Uword           failed;
Uword           i;
Error           error;
...

info.rtNo       = 1;
info.rxCommand  = 0x082A;
info.txCommand  = 0x0C2A;
info.rxBcast    = 0xF82A;
info.txData     = NO_PTR;

for(i=0; i < VP_TOT_5222; i++)
{
   info.sbits[i] = VP_DO_TEST;
}

error = vp5222(&card1, &info, &failed);
```

4.12. vp5223()

Function:-

Error vp5223(vpHandle *cardHandle, vp5223Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp5223Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 5223 test failed for any reason.

Description:-

Function vp5223() carries out the test requirements of paragraph 5.2.2.3 of the RT Validation Test Plan on the UUT. The values in the vp5223Info structure are for specifying various options and variables and reporting back detailed result information.

The vp5223Info structure is:

```
typedef struct
{
   Uword  rtNo;
   Uword  txCommand;
   Uword  illegalRX;
   Uword  illegalTX;
   Uword  txLast;
}vp5223Info;
```

The elements of the vp5223Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30)

txCommand:
This shall be set to define the desired TX command to be used for the 5223 test.

illegalRX:
Illegal RX command to use for test.

illegalTX:
Illegal TX command to use for test.

txLast:
Flag set TRUE by user if TX LAST COMMAND mode code is to be used.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle        card1;
vp5223Info      info;
Uword           failed;
Error           error;
...

info.rtNo       = 1;
info.txCommand  = 0x0C2A;
info.illegalRX  = 0x086A;
info.illegalTX  = 0x0C6A;
info.txLast     = 1;

error = vp5223(&card1, &info, &failed);
```

4.13. vp5224()

Function:-

Error vp5224(vpHandle *cardHandle, vp5224Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp5224Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 5224 test failed for any reason.

Description:-
Function vp5224() carries out the test requirements of paragraph 5.2.2.4 of the RT Validation Test Plan on the UUT. The values in the vp5224Info structure are for specifying various options and variables and reporting back detailed result information.

The vp5224Info structure is:

```
typedef struct
{
   Uword  rtNo;
   Uword  rxCommand;
   Uword  txCommand;
   Uword  priID;
   Uword  secID;
   Uword  modes[VP_TOT_5224];
   Uword  modeXXXXX;
   Uword  *txData;
   Uword  results[VP_TOT_5224];
}vp5224Info;
```

The elements of the vp5224Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30)

modes:
Array defining which mode codes require testing.

t**xCommand:**
This shall be set to define the desired TX command to be used for the 5224 test.

rxCommand:
This shall be set to define the desired RX command to be used for the 5224 test.

priID:
Word to be used by the SELECTIVE SHUTDOWN modes to define the primary bus.

secID:
Word to be used by the SELECTIVE SHUTDOWN modes to define the secondary bus.

modeXXXXX:
Mode code bit pattern to use (00000 or 11111).

***txData:**
TX data for test equipment (can be set to NO_PTR for default data).

results[]:
Results of 5224 test.

Note:

- The 'modes' array have the following elements:
  VP_BCDBCA_TEST - Broadcast Dynamic bus control acceptance
  VP_BCSYNCWO_TEST - Broadcast Synchronise without data word
  VP_BCSYNCWI_TEST - Broadcast Synchronise with data word
  VP_BCSTEST_TEST - Broadcast Initiate selftest
  VP_BCSHUTX_TEST - Broadcast Transmitter shutdown
  VP_BCSELTX_TEST - Broadcast Selective transmitter shutdown
  VP_BCTINHB_TEST - Broadcast Transmit inhibit
  VP_BCRESET_TEST - Broadcast Terminal reset
  Each one of these elements must be initialised to VP_NO_TEST or VP_DO_TEST.
- This function should be executed twice. Once with modeXXXXX = 0x00 and once with modeXXXXX = 0x1F.
- The results array have the same elements as the 'modes' array. If a `particular mode code is tested and fails, the corresponding results array element will be set TRUE.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle          card1;
vp5224Info        info;
Uword             failed;
Uword             i;
Error             error;
...

info.rtNo            = 1;
info.rxCommand    = 0x082A;
info.txCommand    = 0x0C2A;
info.priID           = 0x0000;
info.secID           = 0x0001;
info.modeXXXXX  = 0x001F;
info.txData          = NO_PTR;

for(i=0; i < VP_TOT_5224; i++)
{
   info.modes[i] = VP_DO_TEST;
}

error = vp5224(&card1, &info, &failed);
```

4.14. vp5225()

Function:-

Error vp5225(vpHandle *cardHandle, vp5225Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp5225Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 5225 test failed for any reason.

Description:-

Function vp5225() carries out the test requirements of paragraph 5.2.2.5 of the RT Validation Test Plan on the UUT. The values in the vp5225Info structure are for specifying various options and variables and reporting back detailed result information.

The vp5225Info structure is:

```
typedef struct
{
   Uword  rtNo;
   Uword  rxCommand;
   Uword  rxBcast;
   Uword  *txData;
   Uword  results[VP_TOT_5225];
}vp5225Info;
```

The elements of the vp5225Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30)

rxCommand:
This shall be set to define the desired RX command to be used for the 5225 test.

rxBcast:
This shall be set to define the desired RX broadcast command to be used for the 5225 test.

***txData:**
TX data for test equipment (can be set to NO_PTR for default data).

results[]:
Results of 5225 test.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle        card1;
vp5225Info      info;
Uword           failed;
Uword           i;
Error           error;
...

info.rtNo       = 1;
info.rxCommand  = 0x082A;
info.rxBcast    = 0xF82A;
info.txData     = NO_PTR;

error = vp5225(&card1, &info, &failed);
```

5. HS FUNCTIONS

5.1. hp541()

Function:-

Error hp541(vpHandle *cardHandle, hp541Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp541Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 541 test failed for any reason.

Description:-

Function hp541() carries out the test requirements of paragraph 5.4.1 of the prEN 3910 RT Validation Test Plan on the UUT. The values in the vp541Info structure are for specifying various options and variables and reporting back detailed result information.

The vp541Info structure is:

```
typedef struct
{
   Uword  rtNo;
   Uword  rxBlocks[128];
   Uword  txBlocks[128];
   Uword  results[VP_TOT_541];
} hp541Info;
```

The elements of the vp541Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30).

rxBlocks[n]:
HS Receive message tests. This array shall define the maximum allowable block count for the corresponding message identifier 'n'. A value of 0 shall indicate that the identifier is illegal.

txBlocks[n]:
HS Transmit message tests. This array shall define the maximum allowable block count or the corresponding message identifier 'n'. A value of 0 shall indicate that the identifier is illegal.

results[]:
Results of 541 test.

results[] shall contain an itemised failure report as follows:-

results[VP_HS_5411]	set if section 5.4.1.1 test failed.
results[VP_HS_5412]	set if section 5.4.1.2 test failed.
results[VP_HS_5413R]	set if section 5.4.1.3 test failed (RT-RT UUT RX).
results[VP_HS_5413T]	set if section 5.4.1.3 test failed (RT-RT UUT TX).

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to 'NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle        card1;
vp541Info       info;
Uword           failed, i;
Error           error;
...

info.rtNo = 1;

for(i=0; i!=128; i++)
{
   info.rxblocks[i] = 128;
   info.txblocks[i] = 128;
}

error = vp541(&card1, &info, &failed);
```

5.2. hp542()

Function:-

Error hp542(vpHandle *cardHandle, hp542Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp542Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 542 test failed for any reason.

Description:-

Function hp542() carries out the test requirements of paragraph 5.4.2 of the prEN 3910 RT Validation Test Plan on the UUT. The values in the vp542Info structure are for specifying various options and variables and reporting back detailed result information.

The vp542Info structure is:

```
typedef struct
{
   Uword  rtNo;
} hp542Info;
```

The elements of the vp542Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30)

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to 'NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle        card1;
vp542Info       info;
Uword           failed;
Error           error;
...

info.rtNo = 1;

error = vp542(&card1, &info, &failed);
```

5.3. hp543()

Function:-

Error hp543(vpHandle *cardHandle, hp543Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp543Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 543 test failed for any reason.

Description:-

Function hp543() carries out the test requirements of paragraph 5.4.3 of the prEN 3910 RT Validation Test Plan on the UUT. The values in the vp543Info structure are for specifying various options and variables and reporting back detailed result information.

The vp543Info structure is:

```
typedef struct
{
   Uword  rtNo;
   Uword  noModes;
   Uword  *hsModes;
   Uword  *results;
} hp543Info;
```

The elements of the vp543Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30).

noModes:
Number of HS mode codes in 'hsModes' to test.

hsModes:
Pointer to array of size 'noModes' containing the different mode codes to test.

results:
Pointer to array of size 'noModes' containing the results of the individual mode code tests. If an element value is TRUE the test failed.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to 'NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle          card1;
vp543Info         info;
Uword             failed, i, modeList[20], testResults[20];
Error             error;
...

info.rtNo         = 1;
info.noModes      = 20;

for(i=0; i!=20; i++)
{
   modeList[i]    = i;
   testResults[i] = 0;
}

info->hsModes = modeList;
info->results = testResults;

error = vp543(&card1, &info, &failed);
```

5.4. hp544()

Function:-

Error hp544(vpHandle *cardHandle, hp544Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp544Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 544 test failed for any reason.

Description:-

Function hp544() carries out the test requirements of paragraph 5.4.4 of the prEN 3910 RT Validation Test Plan on the UUT. The values in the vp544Info structure are for specifying various options and variables and reporting back detailed result information.

The vp544Info structure is:

```
typedef struct
{
   Uword  rtNo;
   Uword  noModes;
   Uword  *hsModes;
   Uword  *addData;
   Uword  *results;
} hp544Info;
```

The elements of the vp544Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30)

noModes:
Number of HS mode codes in 'hsModes' to test.

hsModes:
Pointer to array of size 'noModes' containing the different mode codes to test.

addData:
Pointer to array of size 'noModes' containing the different additional data words for the mode codes. If this value is set to 'NO_PTR' then the function will use its own default data values.

results:
Pointer to array of size 'noModes' containing the results of the individual mode code tests. If an element value is TRUE the test failed.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to 'NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle          card1;
vp544Info         info;
Uword             failed, modeList[20], testResults[20], i;
Error             error;
...

info.rtNo         = 1;
info.noModes      = 20;

for(i=0; i!=20; i++)
{
   modeList[i]    = i;
   testResults[i] = 0;
   info.addData   = 0xAAAA;
}

info->hsModes     = modeList;
info->results     = testResults;
info->txData      = NO_PTR;

error = vp544(&card1, &info, &failed);
```

5.5. hp545()

Function:-

Error hp545(vpHandle *cardHandle, hp545Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp545Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 545 test failed for any reason.

Description:-

Function hp545() carries out the test requirements of paragraph 5.4.5 of the prEN 3910 RT Validation Test Plan on the UUT. The values in the vp545Info structure are for specifying various options and variables and reporting back detailed result information.

The vp545Info structure is:

```
typedef struct
{
   Uword  rtNo;
   Uword  rxBlocks[128];
   Uword  txBlocks[128];
   Uword  noModes;
   Uword  *hsModes;
   Uword  *wcnts;
   Uword  *addData;
   Uword  *mResults;
   Uword  tResults[VP_TOT_545];
} hp545Info;
```

The elements of the vp545Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30)

rxBlocks[n]:
HS Receive message tests. This array shall define the maximum allowable block count for the corresponding message identifier 'n'. A value of 0 shall indicate that the identifier is illegal.

txBlocks[n]:
HS Transmit message tests. This array shall define the maximum allowable block count for the corresponding message identifier 'n'. A value of 0 shall indicate that the identifier is illegal.

noModes:
Number of HS mode codes in 'hsModes' to test.

hsModes:
Pointer to array of size 'noModes' containing the different mode codes to test.

wcnts:
Pointer to array of size 'noModes' containing the different wordcounts for the mode codes. These values MUST be 1 or 2.

addData:
Pointer to array of size 'noModes' containing the different additional data words for the mode codes. If this value is set to 'NO_PTR' then the function will use its own default data values.

mResults:
Pointer to array of size 'noModes' containing the results of the individual mode code tests. If an element value is TRUE the test failed.

tResults[]:
Results of 545 HS data transfer test.

tResults[] shall contain an itemised failure report as follows:-

results[VP_HS_5451] set if section 5.4.5.1 test failed.
results[VP_HS_5452R] set if section 5.4.5.2 test failed (RT-RT UUT RX).
results[VP_HS_5452T] set if section 5.4.5.2 test failed (RT-RT UUT TX).

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to 'NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle          card1;
vp545Info         info;
Uword             failed, i;
Uword             modeList[20], modeCnts[20], modeResults[20];
Error             error;
...

info.rtNo         = 1;
info.noModes      = 20;

for(i=0; i!=128; i++)
{
   info.rxblocks[i]          = 128;
   info.txblocks[i]= 128;
}

for(i=0; i!=20; i++)
{
   modeList[i]    = i;
   modeCnts[i]    = 2;
}

info.addData      = NO_PTR;
info.hsModes      = modeList;
info.mResults     = modeResults;
info.wcnts        = modeCnts;

error = vp545(&card1, &info, &failed);
```

5.6. hp551()

Function:-

Error hp551(vpHandle *cardHandle, hp551Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp551Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 551 test failed for any reason.

Description:-

Function hp551() carries out the test requirements of paragraph 5.5.1 of the prEN 3910 RT Validation Test Plan on the UUT. The values in the vp551Info structure are for specifying various options and variables and reporting back detailed result information.

The vp551Info structure is:

```
typedef struct
{
   Uword  rtNo;
   Uword  rxBlocks;
   Uword  txBlocks;
   Uword  rxIdent;
   Uword  txIdent;
} hp551Info;
```

The elements of the vp551Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30)

rxBlocks:
Number of RX blocks to use.

txBlocks:
Number of TX blocks to use.

rxIdent:
RX identifier to use.

txIdent:
TX identifier to use.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to 'NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle        card1;
vp551Info       info;
Uword           failed, i;
Error           error;
...

info.rtNo       = 1;
info.rxBlocks   = 128;
info.txBlocks   = 128;
info.rxIdent    = 4;
info.txIdent    = 4;

error = vp551(&card1, &info, &failed);
```

5.7. hp552()

Function:-

Error hp552(vpHandle *cardHandle, hp552Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp552Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 552 test failed for any reason.

Description:-

Function hp552() carries out the test requirements of paragraph 5.5.2 of the prEN 3910 RT Validation Test Plan on the UUT. The values in the vp552Info structure are for specifying various options and variables and reporting back detailed result information.

The vp552Info structure is:

```
typedef struct
{
   Uword  rtNo;
   Uword  rxBlocks;
   Uword  txBlocks;
   Uword  rxIdent;
   Uword  txIdent;
   Uword  bCasts;
   Uword  bitWord;
   Uword  twoChan;
   Uword  rxInit;
   Uword  txInit;
   Uword  addMode;
   Uword  addData;
   Uword  results[VP_TOT_552];
} hp552Info;
```

The elements of the vp552Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30)

rxBlocks:
Number of RX blocks to use.

txBlocks:
Number of TX blocks to use.

rxIdent:
RX identifier to use.

txIdent:
TX identifier to use.

bCasts:
Flag set if test if for BROADCAST mode codes. If BROADCASTS are allowed this function should be executed twice, first with bCasts=0, then again with bCasts=1.

bitWord:
Value expected for a good HS bit word.

twoChan:
Set if dual redundant 3910 buses used.

rxInit:
Flag set if RX INIT mode code to be tested.

txInit:
Flag set if TX INIT mode code to be tested.

addMode:
Mode With Additional Data to use for test. If 0 then test is disabled.

addData:
Additional data word for mode test - if required.

results[]:
Results of 552 test.

results[] shall contain an itemised failure report as follows:-

results[VP_HS_55211] set if section 5.5.2.1.1 test failed.
results[VP_HS_55212] set if section 5.4.2.1.2 test failed.
results[VP_HS_55213] set if section 5.4.2.1.3 test failed.
results[VP_HS_55221] set if section 5.4.2.2.1 test failed.
results[VP_HS_55222] set if section 5.4.2.2.2 test failed.
results[VP_HS_55231] set if section 5.4.2.3.1 test failed.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to 'NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle        card1;
vp552Info       info;
Uword           failed, i;
Error           error;
...

info.rtNo       = 1;
info.rxBlocks   = 128;
info.txBlocks   = 128;
info.rxIdent    = 4;
info.txIdent    = 4;
info.bCasts     = 0;
info.bitWord    = 0;
info.twoChan    = 0;
info.rxInit     = 1;
info.txInit     = 1;
info.addMode    = 16;
info.addData    = 0;

error = vp552(&card1, &info, &failed);
```

5.8. hp553()

Function:-

Error hp553(vpHandle *cardHandle, hp553Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp553Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 553 test failed for any reason.

Description:-

Function hp553() carries out the test requirements of paragraph 5.5.3 of the prEN 3910 RT Validation Test Plan on the UUT. The values in the vp553Info structure are for specifying various options and variables and reporting back detailed result information.

The vp553Info structure is:

```
typedef struct
{
   Uword  rtNo;
   Uword  rxBlocks;
   Uword  rxIdent;
   Uword  bCasts;
} hp553Info;
```

The elements of the vp553Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30)

rxBlocks:
Number of RX blocks to use.

rxIdent:
RX identifier to use.

bCasts:
Flag set if test if for BROADCAST mode codes. If BROADCASTS are allowed the this function should be executed twice, first with bCasts=0, then again with bCasts=1.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to 'NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle        card1;
vp553Info       info;
Uword           failed, i;
Error           error;
...

info.rtNo       = 1;
info.rxBlocks   = 128;
info.rxIdent    = 4;
info.bCasts     = 0;

error = vp553(&card1, &info, &failed);
```

5.9. hp554()

Function:-

Error hp554(vpHandle *cardHandle, hp554Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp554Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 554 test failed for any reason.

Description:-

Function hp554() carries out the test requirements of paragraph 5.5.4 of the prEN 3910 RT Validation Test Plan on the UUT. The values in the vp554Info structure are for specifying various options and variables and reporting back detailed result information.

The vp554Info structure is:

```
typedef struct
{
   Uword  rtNo;
   Uword  rxBlocks;
   Uword  txBlocks;
   Uword  rxIdent;
   Uword  txIdent;
   Uword  bCasts;
   Uword  bitWord;
   Uword  illegal;
   Uword  results[VP_TOT_554];
} hp554Info;
```

The elements of the vp554Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30)

rxBlocks:
Number of RX blocks to use.

txBlocks:
Number of TX blocks to use.

rxIdent:
RX identifier to use.

txIdent:
TX identifier to use.

bCasts:
Flag set if test if for BROADCAST mode codes. If BROADCASTS are allowed this function should be executed twice, first with bCasts=0, then again with bCasts=1.

bitWord:
Value expected for a good HS bit word.

illegal:
Illegal ACTION WORD to use in test.

results[]:
Results of 554 test.

results[] shall contain an itemised failure report as follows:-

results[VP_HS_55411] set if section 5.5.4.1.1 test failed.
results[VP_HS_55412] set if section 5.4.4.1.2 test failed.
results[VP_HS_55413] set if section 5.4.4.1.3 test failed.
results[VP_HS_55421] set if section 5.4.4.2.1 test failed.
results[VP_HS_55431] set if section 5.4.4.3.1 test failed.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to 'NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle        card1;
vp554Info       info;
Uword           failed, i;
Error           error;
...

info.rtNo       = 1;
info.rxBlocks   = 128;
info.txBlocks   = 128;
info.rxIdent    = 4;
info.txIdent    = 4;
info.bCasts     = 0;
info.bitWord    = 0;
info.illegal    = 0xAAAA;

error = vp554(&card1, &info, &failed);
```

5.10. hp555()

Function:-

Error hp555(vpHandle *cardHandle, hp555Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp555Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 555 test failed for any reason.

Description:-

Function hp555() carries out the test requirements of paragraph 5.5.5 of the prEN 3910 RT Validation Test Plan on the UUT. The values in the vp555Info structure are for specifying various options and variables and reporting back detailed result information.

The vp555Info structure is:

```
typedef struct
{
   Uword  rtNo;
   Uword  rxBlocks;
   Uword  txBlocks;
   Uword  rxId1;
   Uword  rxId2;
   Uword  txId1;
   Uword  txId2;
   Uword  rxInit;
   Uword  txInit;
   Uword  twoChan;
   Uword  results[VP_TOT_555];
} hp555Info;
```

The elements of the vp555Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30)

rxBlocks:
Number of RX blocks to use.

txBlocks:
Number of TX blocks to use.

rxId1:
One of 2 RX identifiers to use.

rxId2:
One of 2 RX identifiers to use.

txId1:
One of 2 TX identifiers to use.

txId2:
One of 2 TX identifiers to use.

rxInit:
Flag set if RX INIT mode code to be tested

txInit:
Flag set if TX INIT mode code to be tested

twoChan:
Set if dual redundant 3910 buses used.

results[]:
Results of 555 test.

results[] shall contain an itemised failure report as follows:-

results[VP_HS_5551] set if section 5.5.5.1 test failed.
results[VP_HS_5552] set if section 5.4.5.2 test failed.
results[VP_HS_5553] set if section 5.4.5.3 test failed.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to 'NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle        card1;
vp555Info       info;
Uword           failed, i;
Error           error;
...

info.rtNo       = 1;
info.rxBlocks   = 128;
info.txBlocks   = 128;
info.rxId1      = 4;
info.rxId2      = 5;
info.txId1      = 4;
info.txId2      = 5;
info.rxInit     = 1;
info.txInit     = 1;
info.twoChan    = 1;

error = vp555(&card1, &info, &failed);
```

5.11. hp556()

Function:-

Error hp556(vpHandle *cardHandle, hp556Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp556Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 556 test failed for any reason.

Description:-

Function hp556() carries out the test requirements of paragraph 5.5.6 of the prEN 3910 RT Validation Test Plan on the UUT. The values in the vp556Info structure are for specifying various options and variables and reporting back detailed result information.

The vp556Info structure is:

```
ypedef struct

  Uword  rtNo;
  Uword  rxBlocks;
  Uword  txBlocks;
  Uword  rxIdent;
  Uword  txIdent;
  Uword  results[VP_TOT_556];
hp556Info;
```

The elements of the vp556Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30)

rxBlocks:
Number of RX blocks to use.

txBlocks:
Number of TX blocks to use.

rxIdent:
RX identifier number to use.

txIdent:
TX identifier number to use.

results[]:
Results of 556 test.

results[] shall contain an itemised failure report as follows:-

results[VP_HS_5561] set if section 5.5.6.1 test failed.
results[VP_HS_5562] set if section 5.4.6.2 test failed.
results[VP_HS_5563] set if section 5.4.6.3 test failed.
results[VP_HS_5564] set if section 5.4.6.4 test failed.
results[VP_HS_5565] set if section 5.4.6.5 test failed.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to 'NO_ERROR'.
If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle        card1;
vp556Info       info;
Uword           failed, i;
Error           error;
...

info.rtNo       = 1;
info.rxBlocks   = 128;
info.txBlocks   = 128;
info.rxIdent    = 4;
info.txIdent    = 4;

error = vp556(&card1, &info, &failed);
```

5.12. hp561()

Function:-

Error hp561(vpHandle *cardHandle, hp561Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp561Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 561 test failed for any reason.

Description:-

Function hp561() carries out the test requirements of paragraph 5.6.1 of the prEN 3910 RT Validation Test Plan on the UUT. The values in the vp561Info structure are for specifying various options and variables and reporting back detailed result information.

The vp561Info structure is:

```
typedef struct
{
  Uword  rtNo;
  Uword  illegalModeTest;
  Uword  illegalMode;
  Uword  addData;
  Uword  rxBlocks;
  Uword  txBlocks;
  Uword  rxIdent;
  Uword  txIdent;
  Uword  illegalIdent[128];
  Uword  illegalBlocks[128];
  Uword  results[VP_TOT_561];
 hp561Info;
```

The elements of the vp561Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30)

illegalModeTest:
Illegal mode code wordcount 0,1 or 2 (0 = No Test).

illegalMode:
Illegal mode code for test.

addData:
Additional data word for mode code.

rxBlocks:
Number of RX blocks to use.

txBlocks:
Number of TX blocks to use.

rxIdent:
RX identifier number to use.

txIdent:
TX identifier number to use.

illegalIdent[]:
Array of Illegal identifiers to use. If illegalIdent[IDno] is TRUE then IDno is illegal.

illegalBlocks[]:
Array of Illegal block counts to use. If illegalBlockst[BKno] is TRUE then BKno is illegal.

results[]:
Results of 561 test.

Note:

- If the element illegalModeTest is 0 then the function shall test 56112, 56113 and 56121. If illegalModeTest is 1 or 2 the just a single illegal mode code defined by illegalMode shall be sent to enable the testing of section 56111. In this case illegalModeTest shall define the command wordcount to be used (2 is for mode with additional data word).

results[] shall contain an itemised failure report as follows:-

results[VP_HS_56111] set if section 5.6.1.1.1 test failed.
results[VP_HS_56112] set if section 5.6.1.1.2 test failed.
results[VP_HS_56113] set if section 5.6.1.1.3 test failed.
results[VP_HS_56121] set if section 5.6.1.2.1 test failed.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to 'NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle          card1;
vp561Info         info;
Uword             failed, i;
Error             error;
...

info.rtNo                = 1;
info.rxBlocks            = 128;
info.txBlocks            = 128;
info.rxIdent             = 4;
info.txIdent             = 4;
info.addData             = 0xAAAA;
info.illegalModeTest  = 0;
info.illegalMode         = 0;

for(i=0; i!=128; i++)
{
   if(i < 44)
      info.illegalIdent[i] = 0;
   else
      info.illegalIdent[i] = 1;

   if(i < 108)
      info.illegalBlocks[i] = 0;
   else
      info.illegalBlocks[i] = 1;
}

error = vp561(&card1, &info, &failed);
```

5.13. hp562()

Function:-

Error hp562(vpHandle *cardHandle, hp562Info *info, Uword *failed);

PARAMETER	DESCRIPTION
cardHandle	This is a pointer to an uninitialised vpHandle structure.
info	Pointer to a vp562Info structure defining variables for test.
failed	Pointer to Uword for storage of test result. This value will be set TRUE if the 562 test failed for any reason.

Description:-

Function hp562() carries out the test requirements of paragraph 5.6.2 of the prEN 3910 RT Validation Test Plan on the UUT. The values in the vp562Info structure are for specifying various options and variables and reporting back detailed result information.

The vp562Info structure is:

```
  Uword  rtNo;
  Uword  rxBlocks;
  Uword  txBlocks;
  Uword  rxIdent;
  Uword  txIdent;
  Uword  addMode;
  Uword  addData;
  Uword  results[VP_TOT_562];
hp562Info;
```

The elements of the vp562Info structure should be set as follows :-

rtNo:
Address of UUT (0 to 30).

rxBlocks:
Number of RX blocks to use.

txBlocks:
Number of TX blocks to use.

rxIdent:
RX identifier number to use.

txIdent:
TX identifier number to use.

addMode:
Mode code with additional data word.

addData:
Additional data word for mode code.

results[]:
Results of 562 test.

results[] shall contain an itemised failure report as follows:-

results[VP_HS_56211] set if section 5.6.2.1.1 test failed.
results[VP_HS_56212] set if section 5.6.2.1.2 test failed.
results[VP_HS_56213] set if section 5.6.2.1.3 test failed.
results[VP_HS_56221] set if section 5.6.2.2.1 test failed.
results[VP_HS_56222] set if section 5.6.2.2.2 test failed.
results[VP_HS_56223] set if section 5.6.2.2.3 test failed.
results[VP_HS_56224] set if section 5.6.2.2.4 test failed.
results[VP_HS_56225] set if section 5.6.2.2.5 test failed.
results[VP_HS_56226] set if section 5.6.2.2.6 test failed.
results[VP_HS_56227] set if section 5.6.2.2.7 test failed.
results[VP_HS_56228] set if section 5.6.2.2.8 test failed.

Return Value:-

A numeric parameter of type Error is returned. If all input parameters are valid this is returned equal to 'NO_ERROR'. If an invalid parameter is used, this will be returned equal to one of the values defined in the errors list (see appendix).

Example:-

```
#include  "vplan.h"
...
vpHandle        card1;
vp562Info       info;
Uword           failed, i;
Error           error;
...

info.rtNo       = 1;
info.rxBlocks   = 128;
info.txBlocks   = 128;
info.rxIdent    = 4;
info.txIdent    = 4;
info.addMode    = 0x0010;
info.addData    = 0xAAAA;

error = vp562(&card1, &info, &failed);
```

6. APPENDIX A

List of function:

Error vpInit(vpHandle *cardHandle, Ulong base);
Error vpInitPC(vpHandle *cardHandle, vpPCinfo *info);

Error vp5211(vpHandle *cardHandle, vp5211Info *info, Uword *failed);
Error vp5212(vpHandle *cardHandle, vp5212Info *info, Uword *failed);
Error vp5213(vpHandle *cardHandle, vp5213Info *info, Uword *failed);
Error vp5214(vpHandle *cardHandle, vp5214Info *info, Uword *failed);
Error vp5215(vpHandle *cardHandle, vp5215Info *info, Uword *failed);
Error vp5216(vpHandle *cardHandle, vp5216Info *info, Uword *failed);
Error vp5217(vpHandle *cardHandle, vp5217Info *info, Uword *failed);
Error vp5218(vpHandle *cardHandle, vp5218Info *info, Uword *failed);
Error vp5219(vpHandle *cardHandle, vp5219Info *info, Uword *failed);
Error vp5221(vpHandle *cardHandle, vp5221Info *info, Uword *failed);
Error vp5222(vpHandle *cardHandle, vp5222Info *info, Uword *failed);
Error vp5223(vpHandle *cardHandle, vp5223Info *info, Uword *failed);
Error vp5224(vpHandle *cardHandle, vp5224Info *info, Uword *failed);
Error vp5225(vpHandle *cardHandle, vp5225Info *info, Uword *failed);

Error hp541(vpHandle *cardHandle, hp541Info *info, Uword *failed);
Error hp542(vpHandle *cardHandle, hp542Info *info, Uword *failed);
Error hp543(vpHandle *cardHandle, hp543Info *info, Uword *failed);
Error hp544(vpHandle *cardHandle, hp544Info *info, Uword *failed);
Error hp545(vpHandle *cardHandle, hp545Info *info, Uword *failed);
Error hp551(vpHandle *cardHandle, hp551Info *info, Uword *failed);
Error hp552(vpHandle *cardHandle, hp552Info *info, Uword *failed);
Error hp553(vpHandle *cardHandle, hp553Info *info, Uword *failed);
Error hp554(vpHandle *cardHandle, hp554Info *info, Uword *failed);
Error hp562(vpHandle *cardHandle, hp562Info *info, Uword *failed);
Error hp562(vpHandle *cardHandle, hp562Info *info, Uword *failed);
Error hp562(vpHandle *cardHandle, hp562Info *info, Uword *failed);
Error hp562(vpHandle *cardHandle, hp562Info *info, Uword *failed);

7. APPENDIX B

List of error numbers returned by the functions:

The error numbers returned by the functions are the decimal equivalent of the RT and prEN 3910 Validation Test plans paragraph where the error was detected as follows:-

General Errors

Error name	Decimal value
NO_ERROR	0
VP_ERR_SETUP	52000
VP_ERR_COMMAND	52001

Ls Errors

Error name	Decimal value
VP_ERR_5211	52110
VP_ERR_52111	52111
VP_ERR_52112	52112
VP_ERR_5212	52120
VP_ERR_52121	52121
VP_ERR_52122	52122
VP_ERR_5213	52130
VP_ERR_52131	52131
VP_ERR_52132	52132
VP_ERR_52133	52133
VP_ERR_52134	52134
VP_ERR_52135	52135
VP_ERR_52136	52136
VP_ERR_52137	52137
VP_ERR_5214	52140
VP_ERR_5215	52150
VP_ERR_5216	52160
VP_ERR_5217	52170
VP_ERR_5218	52180
VP_ERR_5219	52190
VP_ERR_5221	52210
VP_ERR_52216	52216
VP_ERR_5222	52220
VP_ERR_52221	52221
VP_ERR_52223	52223
VP_ERR_52224	52224
VP_ERR_52225	52225
VP_ERR_5223	52230
VP_ERR_5224	52240
VP_ERR_52246	52246
VP_ERR_5225	52250

HS Errors

Error name	Decimal value
VP_ERR_541	54100
VP_ERR_542	54200
VP_ERR_543	54300
VP_ERR_544	54400
VP_ERR_545	54500
VP_ERR_551	55100
VP_ERR_552	55200
VP_ERR_553	55300
VP_ERR_554	55400
VP_ERR_562	56200
VP_ERR_562	56200
VP_ERR_562	56200
VP_ERR_562	56200