5 TurboIMAGE/XL Library Procedures

This chapter contains the reference specifications for the TurboIMAGE/XL library procedures (also known as intrinsics), arranged alphabetically. Table 5-1. gives a summary of the procedures with a brief description of their function.

Using TurboIMAGE/XL Intrinsics

On the following pages, the calling parameters for each procedure are defined in alphabetical order for easy look-up rather than the order in which they appear in the call statement. Every parameter must be included when a call is made because a parameter's meaning is determined by its position.

NOTE All parameters must be on halfword boundaries. Database names, data set names, and data item names that are passed to the TurboIMAGE/XL intrinsics must be in uppercase.

Table 5-1. TurboIMAGE/XL Procedures

Procedures	Function
DBBEGIN	When logging, designates the beginning of a transaction and optionally writes user information in the log file.
DBCLOSE	Terminates access to a database or a dataset, or resets the pointers of a data set to their original state
DBCONTROL*	Allows a process operating in exclusive mode to enable or disable the deferred output (AUTODEFER) option. Also allows a process to enable or disable the critical item update (CIUPDATE) option, activate deadlock detection, and set wildcard character and BTREEMODE1 option for database, or for the current DBOPEN without affecting other processes operating on the same database.
DBDELETE	Deletes an existing entry from data set.
DBEND	When logging, designates the end of a transaction and optionally writes user information to the log file.
DBERROR	Supplies an ASCII language message that interprets the status information set by any callable TurboIMAGE/XL procedure. The message is returned to the calling program in a buffer.
DBEXPLAIN	Examines status information returned by a TurboIMAGE/XL procedure and prints a multilane message on the $STDLIST device.
DBFIND*	Locates the first and last entries of a data chain in preparation for access to entries in the chain for non-B-Tree searches. For B-Tree searches, master and detail data sets can be included.

DBGET	Reads the data item values of a specified entry.
DBINFO*	Provides information about the database being accessed, such as the name and description of a data item or data set. It also provides information on logging, including logging of dynamic and multiple database transactions, and on third party indexing, critical item update, and other options.
DBLOCK	Locks one or more data entries, a data set, or an entire database (or a combination of these) temporarily to allow the process calling the procedure to have exclusive access to the locked entities.
DBMEMO*	When logging, writes user information to the log file.
DBOPEN	Initiates access to a database. Specifies the user access mode and user class number for the duration of the process.
DBPUT*	Adds new entry to a manual master or detail data set.
DBUNLOCK	Releases those locks obtained with previous call(s) to DBLOCK.
DBUPDATE*	Modifies the values of data items. Cannot be used to update master data set key items. Can be used to update detail data set search and sort items in database access mode 1, 3, or 4 if permitted by the critical item update (CIUPDATE) option settings for the database and the current process.
DBXBEGIN	Designates the beginning of a dynamic transaction. Refer to "Transactions" later in this chapter for a description of dynamic transactions.
DBXEND	Designates the end of a dynamic transaction.
DBXUNDO	Rolls back the active dynamic transaction.

Table 5-2. illustrates the forms of the call statements for the languages that can be used to call the procedures. Chapter 6 contains examples of using the procedures and specifications for declaration of parameters for some of these languages. It also provides a sample RPG program.

* These items have OPENTURBO specific features. See each 'Call' section for further details.

Table 5-2. Calling a TurboIMAGE/XL Procedure

COBOL II	`CALL` "*name*" `USING` *parameter,parameter,...,parameter.*
FORTRAN 77	`CALL` *name* (*parameter,parameter,...,parameter*)
Pascal	*name* (*parameter,parameter,...,parameter*);
BBASIC	*linenumber* `CALL` *name* (*parameter,parameter,...,parameter*)
C	*name* (*parameter,parameter,...,parameter*);

All procedures can be called from programs in any of the host languages.

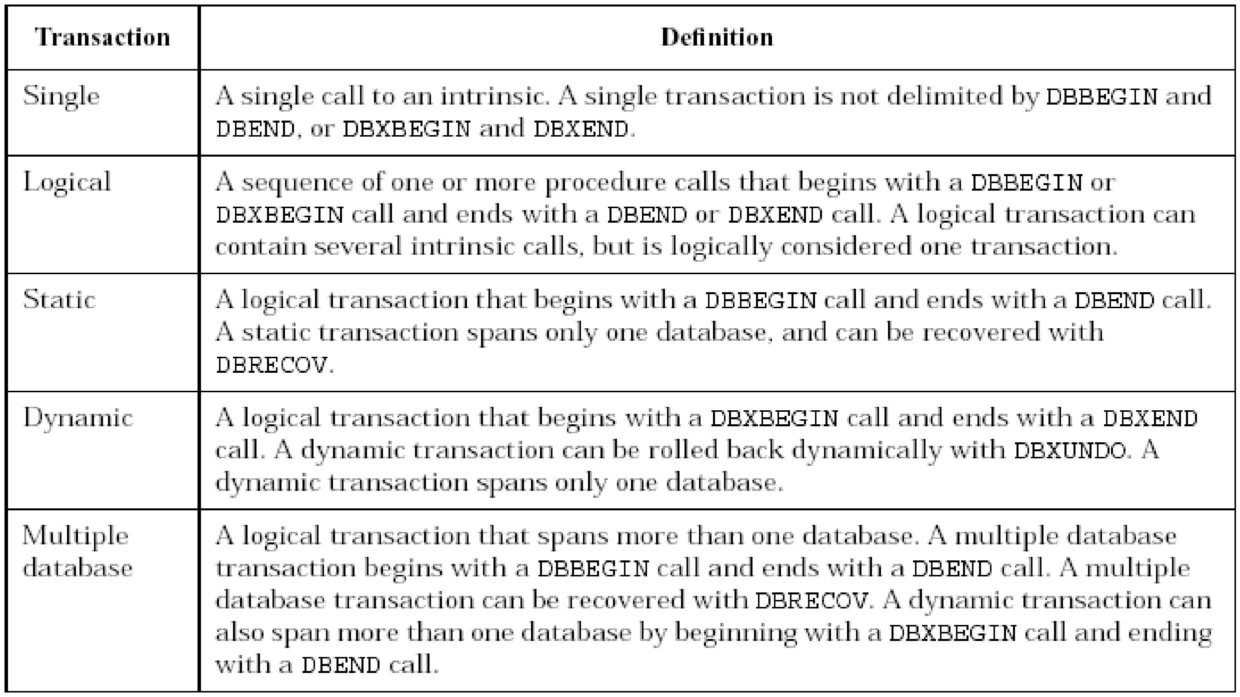
Intrinsic Numbers
An intrinsic number is provided for each procedure. This number, which uniquely identifies the procedure within TurboIMAGE/XL and the MPE/iX operating system, is returned with other status information when an error occurs. You can use it to identify the procedure that caused the error or call `DBEXPLAIN` to interpret the number and other information.

Database Protection
When each procedure is called, TurboIMAGE/XL verifies that the requested operation is compatible with the user class number and access mode established when the database is opened.

Unused Parameters
When calling some procedures for a specific purpose, one of the parameters can be ignored; however, it still must be listed in the call statement. An application program may find it useful to set up a variable named `Not_Used_Parm` or `DUMMY` to be listed as the unused parameter as a reminder that the value of the parameter does not affect the procedure call. Refer to the examples in chapter 6.

The Status Array
The status array is a communication area. If the procedure executes successfully, the contents of the array reflect this as described under each intrinsic discussion in this chapter. If the procedure fails, standard error information is returned in the array as described in this chapter and appendix A.

Transactions
TurboIMAGE/XL transactions are defined below:



Transaction	Definition
Single	A single call to an intrinsic. A single transaction is not delimited by DBBEGIN and DBEND, or DBXBEGIN and DBXEND.
Logical	A sequence of one or more procedure calls that begins with a DBBEGIN or DBXBEGIN call and ends with a DBEND or DBXEND call. A logical transaction can contain several intrinsic calls, but is logically considered one transaction.
Static	A logical transaction that begins with a DBBEGIN call and ends with a DBEND call. A static transaction spans only one database, and can be recovered with DBRECOV.
Dynamic	A logical transaction that begins with a DBXBEGIN call and ends with a DBXEND call. A dynamic transaction can be rolled back dynamically with DBXUNDO. A dynamic transaction spans only one database.
Multiple database	A logical transaction that spans more than one database. A multiple database transaction begins with a DBBEGIN call and ends with a DBEND call. A multiple database transaction can be recovered with DBRECOV. A dynamic transaction can also span more than one database by beginning with a DBXBEGIN call and ending with a DBEND call.

Refer to chapters 4 and 7 for more information on transactions.