$BUILD - Compiler Dialogue Build Utility

1. Introduction

The $BUILD utility allows compiler options, copy-libraries etc. to be included within source programs rather than relying on job streams, $DEV options, a System Rebuilder utility, Packaging Notes (or in some cases memory) to accurately run a compilation.

Although $BUILD is closely associated with the $SDL32 compiler it can be used with other compilers: $SDL and $COBOL (and $XREF). For GSM SP-23, and later, $BUILD can be used with the $COMPILE compiler.

Further "advanced" features (i.e. conditional options) will allow $BUILD to be used internally to replace large parts of the GSM Packaging System.

2. Design Overview

$BUILD prompts for the name and unit of the source file to be compiled. Once the filename and been accepted and the file has been opened a number of options may be specified. For example:

```
Name of file to build:source  Unit:uuu
Option:option-1
Option:option-2
…
Option:<CR>
```

No options may be required in which case the generalised dialogue above degenerates to:

```
Name of file to build:source  Unit:uuu
Option:<CR>
```

File source must be a text (TF) file. A default extension of S. should be assumed.

$BUILD scans the text file looking for the following key-strings. **Each key-string must occur at the start of a newline**:

Key string	Parameter format	Single/multiple occurrences
*#BUILD	:compiler	single

```
*#ENDBUILD                                    single
*#OBJECT       :unit          :size                 single
*#DICT         :dict          :unit                 multiple
*#LIST              :unit                           single
*#LIST              :NONE                     single       (to
suppress listing)
*#LIST              :unit          :size                 single
*#COPY         :file          :unit                 multiple
*#LINK         :library  :unit                     multiple
*#DLM          :member   :library  :unit           multiple
*#OPTION       :string                             multiple
*#VERSION      :string                             single
*#PRE               :program      :param1      :param2      :param3
     multiple
*#RUN          :program  :param1  :param2  :param3     multiple
*#CHAIN        :file :unit                         single
*#OPT          :option                          multiple
*#NOPT         :option                          multiple
*#HOLDING      :file                               single
*#LIBRARY      :library  :unit     :file     :unit          single
*#LIBX         :library  :unit     :file     :unit          single
```

The *#HOLDING option is reserved for GSM packaging. **NOT YET IMPLEMENTED.**

The *#LIBRARY option is only supported by GSM SP-23, and later.

The *#LIBX option is only supported by GSM SP-26, and later.

The *#DLM option is only supported by GSM SP-26, and later.

The "single/multiple occurrences" column indicates whether a $BUILD keyword can appear multiple times in a source file. In general, the 2nd and subsequent occurrences of a single-instance keyword (e.g. *#OBJECT) are ignored by $BUILD.

The following "advanced" options are included in the initial release of $BUILD:

- *#IF option
- *#IFN option
- *#ELSE
- *#ENDIF
- *#FI

3. Generic functionality for $SDL32, $SDL & $COBOL

The first character for each key-string is treated as a comment by all the compilers. The 2nd character allows the key-strings to be differentiated from "normal" comments.

The *#BUILD directive allows the name of the actual compiler to be specified making each source completely self-referencing. The following arguments are allowed:

```
     :$SDL32        (this will select either $SDL32 or $COMPILE,
see section 4.1)
     :$SDL
     :$COBOL
     :$COMPILE      (GSM SP-23, or later)
```

Note that the 16-bit $XREF utility must be invoked using the "XREF" run-time option (see below).

The *#ENDBUILD directive defines the end of the directive block. The *#ENDBUILD directive prevents $BUILD from scanning through to the end of the file and will speed-up pre-processor time.

The syntax of the single *#OBJECT option is the same for all compilers. In all cases, both an object unit and size are required. In the $BUILD parser, the size argument is optional. If a size is not supplied, a value of 0 is generated by $BUILD. If the *#OBJECT option is not specified, $BUILD generates two <CR> replies.

The syntax of the single *#LIST option differs between the various compilers (see below).

The multiple *#DICT options are only valid if the compiler is $SDL or $SDL32 (see below).

The job management generated by the multiple *#COPY options differs between $SDL32/$SDL and $COBOL/$XREF (see below).

The multiple *#LINK options are only valid if the compiler is $SDL or $SDL32/$COMPILE (see below).

The single *#VERSION option is only valid if the compiler is $SDL or $SDL32/$COMPILE (see below).

Any number of *#OPTION options are allowed. These strings are not validated by $BUILD but are passed directly to the compiler's OPTION: prompt. Note that a hard-coded option, specified using the *#OPTION keyword, may be over-ridden by a run-time $BUILD option (see below).

Any number of *#RUN options can be used to add extra free-format job dialogue to the end of the compilation stream (e.g. to run $LINK after using $COBOL, or to copy an object file to another holding unit).

Any number of *#PRE options can be used to add extra free-format job dialogue to the **start** of the compilation stream (i.e. before the compiler dialogue). Note that this option can't be used to run $MACRO on an pre-macro F.xxxxxx source as the primary source (e.g. F.XXSAAA) rather than a derived source (e.g. S.X-SAAA) is used for the compilation stream.

The *#CHAIN option can be used to string $BUILD compilations (i.e. *#CHAIN will invoke another $BUILD specifying the source file and unit).

The *#OPT and *#NOPT options are only useful when used with the *#CHAIN option. They allow run-time options to be passed to the CHAIN'ed $BUILD dialogue.

4. Specific functionality for $SDL32 & $COMPILE

The syntax of the single *#LIST option for $SDL32/$COMPILE compilations requires both a listing unit and a file-size. In the $BUILD parser, the size argument is optional. If a size is not supplied, a value of 0 is generated by $BUILD. If the *#LIST option is not specified, $BUILD generates two <CR> replies.

The file-size is suppressed if the printer unit is a physical printer (the same algorithm that is used in $SDL32/$COMPILE is employed in $BUILD).

Note that the *#LIST option can be suppressed by the NOPR run-time argument. For GSM SP-30, or later, the special parameter “:NONE” (note the leading “:”) can be used to suppress the listing file.

The multiple *#DICT options are only valid if the compiler is $SDL32, $COMPILE or $SDL (see below). Both the dictionary name and unit must be specified. For GSM SP-31, and later, up to 20 *#DICT statements are supported.

Both the copy library and unit must be specified to the multiple *#COPY options. If one, or more, *#COPY options have been specified, $BUILD inserts a response of "COP" to the $SDL32/$COMPILE OPTION: prompt.

Both the DLM library and unit must be specified to the multiple *#LINK options. If one, or more, *#LINK options have been specified, $BUILD inserts a response of "LNK" to the $SDL32/$COMPILE OPTION: prompt.

To specify a single member from a DLM library the *#DLM option, rather than *#LINK, must be used. The library-member, the DLM library and unit must **all** be specified to the multiple *#DLM options. If one, or more, *#DLM options have been specified, $BUILD inserts a response of "LNK" to the $SDL32/$COMPILE OPTION: prompt. The *#DLM option(s) can be mixed with the *#LINK option(s).

If the *#VERSION option has been specified, $BUILD inserts a response of "VER" to the $SDL32/$COMPILE OPTION: prompt.

4.1 Switching $SDL32 to $COMPILE

For GSM SP-23, and later, $COMPILE is recognized as an alternative to $SDL32 as an argument to the *#BUILD directive. However, it may not be convenient to edit a large number of source files to change this compiler directive. For GSM SP-23 (and later) the following registry setting, in the “Customisations” registry key, can be used to direct $BUILD to substitute “$SDL32” by “$COMPILE”:

```
Map$SDL32To$COMPILEin$BUILDDialogue=On
```

This feature requires the use of GLOBAL.EXE V4.2a, or later.

5. Specific functionality for $SDL

The syntax of the single *#LIST option for $SDL compilations requires both a listing unit and a file-size. In the $BUILD parser, the size argument is optional. If a size is not supplied, a value of 0 is generated by $BUILD. If the *#LIST option is not specified, $BUILD generates two <CR> replies.

The file-size is suppressed if the printer unit is a physical printer (the same algorithm that is used in $SDL is employed in $BUILD).

Note that the *#LIST option can be suppressed by the NOPR run-time argument.

The multiple *#DICT options are only valid if the compiler is $SDL or $SDL32 (see above). Both the dictionary name and unit must be specified.

Both the copy library and unit must be specified to the multiple *#COPY options. If one, or more, *#COPY options have been specified, $BUILD inserts a response of "COP" to the $SDL OPTION: prompt. If the filename starts with "S." prefix, the prefix is removed when building the $SDL dialogue.

Both the object library (or object file) and unit must be specified to the multiple *#LINK options. If one, or more, *#LINK options have been specified, $BUILD inserts a response of "LNK" to the $SDL OPTION: prompt. If the filename starts with "C." prefix, the prefix is removed when building the $SDL dialogue.

The *#VERSION option is illegal for the $SDL compiler option.

6. Specific functionality for $COBOL

The syntax of the single *#LIST option for $COBOL compilations only requires a listing unit (i.e. a file-size is not required). If the *#LIST option is not specified, $BUILD generates a <CR> reply.

Note that the *#LIST option can be suppressed by the NOPR run-time argument.

The *#DICT option is illegal for the $COBOL compiler option.

Both the copy library and unit must be specified to the multiple *#COPY options.

The *#LINK option is illegal for the $COBOL compiler option.

The *#VERSION option is illegal for the $COBOL compiler option.

7. Run-time Options

$BUILD recognises the following run-time options:

opt	Specify a compiler option explicitly (e.g. BL, SD etc.). The options specified with the *#OPTION

directives are logically OR'ed with the options supplied at $BUILD run-time;

-opt Turn off a compiler option that is specified using *#OPTION statement (no action if the option is NOT defined);

VER Allow the *#VERSION option to be over-ridden;

NOPR Suppress the generation of the listing file;

XREF replace $COBOL by $XREF (and ignores the *#OBJECT option);

NLIB Suppress the dialogue that would normally be generated by the *#LIBRARY directive (GSM SP-23, or later) or *#LIBX directive (GSM SP-26, or later).

NRUN Suppress the dialogue that would normally be generated by the *#RUN directive (GSM SP-26, or later).

NCHN Suppress the dialogue that would normally be generated by the *#CHAIN directive (GSM SP-26, or later).

The following run-time options are reserved for future use:

"0" Only consider *#CHAIN. *#RUN or *#PRE pre-processor statements (i.e. by-pass any compilation statements). **NOT YET IMPLEMENTED;**

"1" Ignore any *#CHAIN, *#RUN or *#PRE pre-processor statements (i.e. just perform a single compilation). **NOT YET IMPLEMENTED;**

"A" to "Z" Set an internal conditional flag (which may be actioned by a *#IF or *#IFN etc. pre-processor statement). IT IS ASSUMED THAT NO COMPILER DIRECTIVE WILL EVER BE A SINGLE CHARACTER. **NOT YET IMPLEMENTED;**

BA$$ Reserved to compile BA$xxx DLM's. **NOT YET IMPLEMENTED;**

PACK Reserved for GSM packaging. **NOT YET IMPLEMENTED;**

8. Conditional Options

The final part of the $BUILD implementation should include the support of conditional options via the *#IF etc. keywords. Further design is required. **NOT YET IMPLEMENTED**.