**Standard Operation Procedure for Automations of SAP GUI in Visual Basic for Applications**

**10 March, 2017**

**TS-TLP, HAESL**

Table of Contents

[1. Introduction 2](#_Toc477006549)

[2. Working Principle 3](#_Toc477006550)

[3. How to Use 4](#_Toc477006551)

[i. Importing and executing the codes 5](#_Toc477006552)

[ii. Marcos in “Core” Modules 6](#_Toc477006553)

[a. SQ01: Operation finder 6](#_Toc477006554)

[b. SQ01/ZIP11/IP03: Plan details viewer 7](#_Toc477006555)

[c. IA06/IW32/ZL07: operations selector/un-selector 9](#_Toc477006556)

[d. IA06: GotoPlan 11](#_Toc477006557)

[e. IA06/IW32/ZL07: long text copier & paster 13](#_Toc477006558)

[f. Excel: Converting copied operation into vertical form 14](#_Toc477006559)

[g. IA06: Long Text Replacer 16](#_Toc477006560)

[h. ZL07: Pending TV operations adder 16](#_Toc477006561)

[i. ZL07: Reference adder 17](#_Toc477006562)

[j. SQ01: Simplified Plan Finder for Bulk Query 18](#_Toc477006563)

[k. Excel: Concatenating Strings 20](#_Toc477006564)

[l. Excel: Comparing values in two columns 21](#_Toc477006565)

[m. Excel: Extracting Rows with certain value 21](#_Toc477006566)

[n. Excel: Highlighting rows 21](#_Toc477006567)

[o. IA06: Transformation-related Marcos 22](#_Toc477006568)

[iii. Marcos in “Auxiliary/Supplementary” Modules 22](#_Toc477006569)

[a. Generating Snapshot & Search Function 22](#_Toc477006570)

[b. Hits, operations and plans counting for the search result table 22](#_Toc477006571)

[c. Comparing two search results 23](#_Toc477006572)

[d. IA17: Counting keywords 23](#_Toc477006573)

[e. ZL07: Printing out WASH/INSPECT PACKs in plans 23](#_Toc477006574)

[iv. Marcos in “Experimental/Outdated/One-off” Modules 24](#_Toc477006575)

# Introduction

This Standard Operation Procedure (SOP) serves as a guide to the use of marcos written in Visual Basic for Applications (VBA) for using in conjunction with SAP GUI. The files are a loose collection of marcos made for daily operations of Task List Planning (TLP) to automatize mundane and non-technical workloads.

VBA in general sense refers to the API using Visual Basic language which allows control of applications developed by Microsoft. VBA editors are provided in every application in Office Suite. In theory the codes can be ran with any host with VBA compiler. However, as most Marcos written is designed to work with Excel exclusively for the ease of data manipulation, this is neither encouraged nor desirable to use them outside Excel as the codes might result in throwing out errors due to missing objects or data. Therefore, if without explicitly stated so, the term VBA described here means VBA for Excel and VBA for Excel only.

Please do note that there are many functions which cannot be ran directly as Marco as MS Excel limits that the executable must not take variables input. As the result, they need to be called through a “callable” function for it properly run in Excel environment. Since the purpose of this SOP is to guide user through the use of Marco without tampering the codes, this SOP will only tackles the callable functions as there is no way to use the non-callable ones without opening an Integrated Development Environment (IDE) and thus it is beyond the scope of this document.

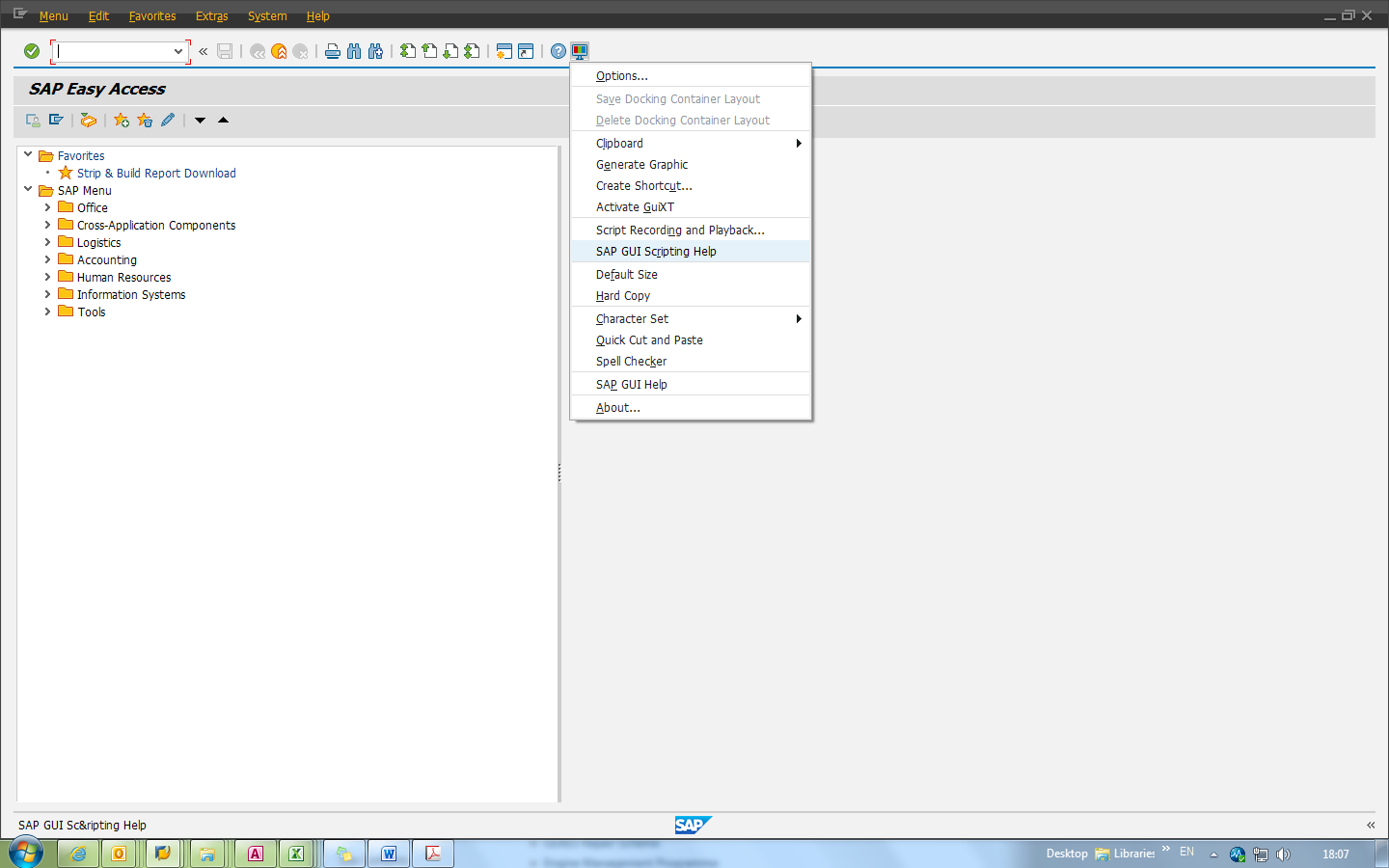
# Working Principle

The Marcos control SAP GUI through an Application Program Interface (API) called “SAP GUI Scripting”. The API controls almost all visual elements visible shown in SAP GUI. By creating related COM objects in VBA, Office applications such as Excel, Access and Word can be used as a host to store instructions which exerts control to SAP GUI.

As the API provides only controls to visual elements, it is not possible to communicate with SAP directly but through SAP GUI. Information is only available when it is shown on screen. Therefore, all Marcos written are essentially imitations of manual user inputs.

For more information about the object model of SAP GUI Scripting, please refer to “SAP GUI Scripting Help”, which is available through any opened SAP GUI window, under the “Customize Local Layout” Icon (Alt +F12).

For more information about the object model of VBA in Excel/Access/Word/Powerpoint/Outlook, please refer to related documents provided by Microsoft.

Fig.1. Location of “SAP GUI Scripting Help”.

# How to Use

To use the programs, the user must have a macro-enabled MS Office Excel to act as host to the codes. Make sure SAP GUI is available – it’s acceptable to run the codes even SAP GUI is currently not opened, but it needs to be able to connect to the Internet and log into the SAP system to carry out the instructions written within the codes.

The codes and related files are stored in [V:\Technical Services\5. Task List Planning\5.2 Team Projects\SAP Marcos\VBA Marco Collection (09 Mar 2017).](file:///V:\Technical%20Services\5.%20Task%20List%20Planning\5.2%20Team%20Projects\SAP%20Marcos\VBA%20Marco%20Collection%20(09%20Mar%202017))

The modules in this collection can be categorized into several groups:

1. Core – Depended by codes in other modules. Contains necessary codes and functions for most of the codes to run properly.
   1. ExcelOperations.bas
   2. Global\_Callables.bas
   3. SAP.bas
2. Auxiliary/Supplementary– Add-on functionality for existing applications. Codes outside of the modules are not dependent of these files.
   1. CompareModule.bas
   2. CountingModule.bas
   3. CountTV.bas
   4. Snapshot\_Dump.bas
   5. The\_Grand\_Search.bas
   6. zl07print\_WASH\_NDT.bas
3. Experimental/Outdated/One-off – Contains outdated/very context-specific or experimental codes that usually will not end nicely when being run without modifying the codes. For references only.
   1. CombiningModule.bas
   2. CruiseModule.bas
   3. HellOnEarth.bas
   4. IE.bas
   5. Operation\_CMM.bas
   6. Operation\_HRC.bas
   7. Operation\_Odyssey.bas
   8. Testing\_Ground.bas
   9. ztext\_tasklistTVcounter.bas

For most purposes, importing only modules in Core and Auxiliary/Supplement will suffice.

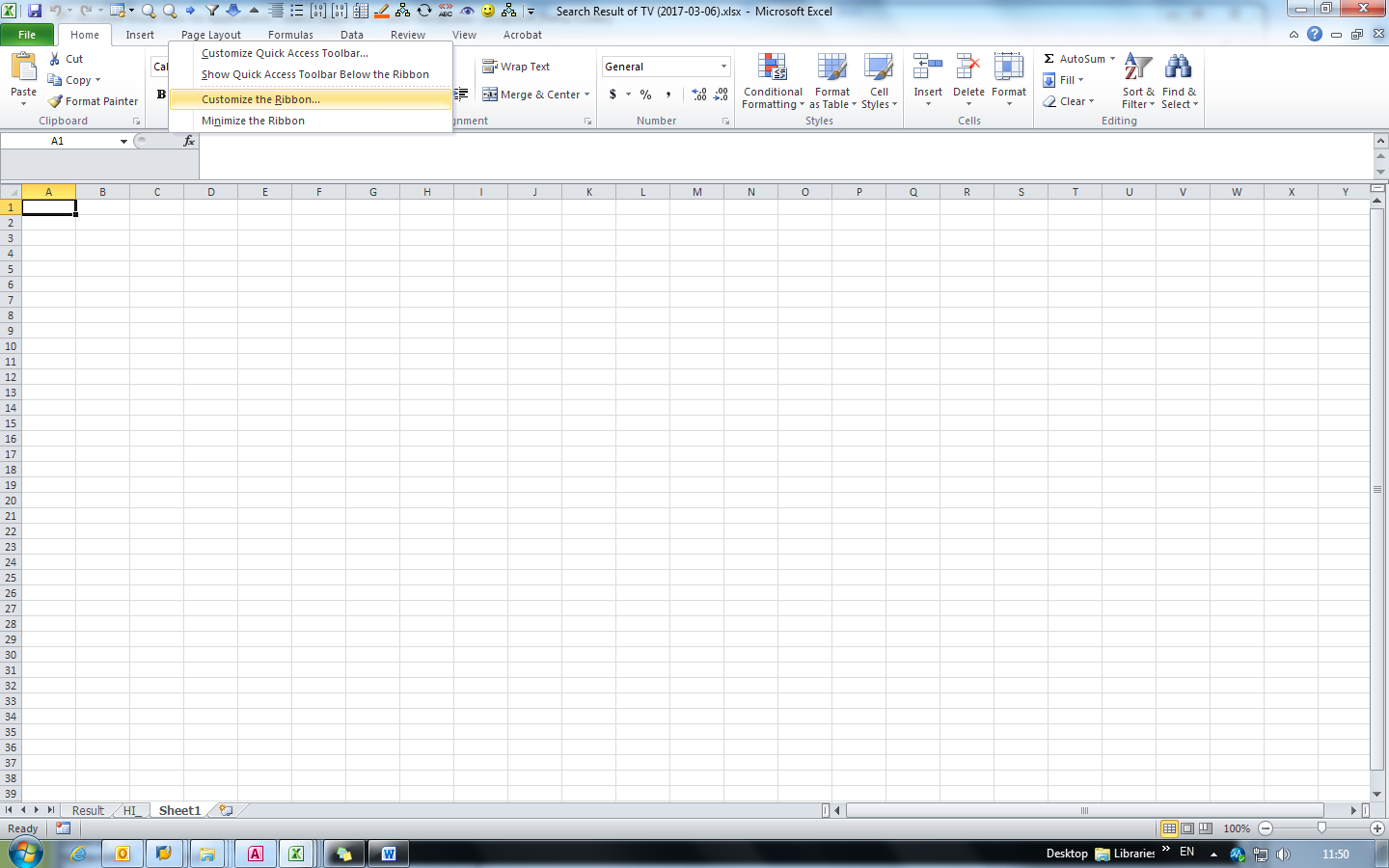
## Importing and executing the codes

There are several formats of file that needs to be imported in order to execute the Marcos:

* .bas file, which contains the source codes.
* .frm file, which contains the customized visual elements used by the codes.
* .cls file, which contains the customized class information used by the codes.

The .frm files and .cls files are only used by a handful of Marcos. The majority of the codes will run without problems even in case that the said .frm and .cls files are not imported.

The codes will need to be imported before being able to run by Excel. This requires importing Visual Basic Exported Module files to Excel’s library. The library can be opened in “Developer” Tab in the Excel ribbons which is hidden by default. To make the tab visible, right-click the ribbon and select “Customize the ribbon…”.

Fig.2. Selecting “Customize the ribbon…”

## Marcos in “Core” Modules

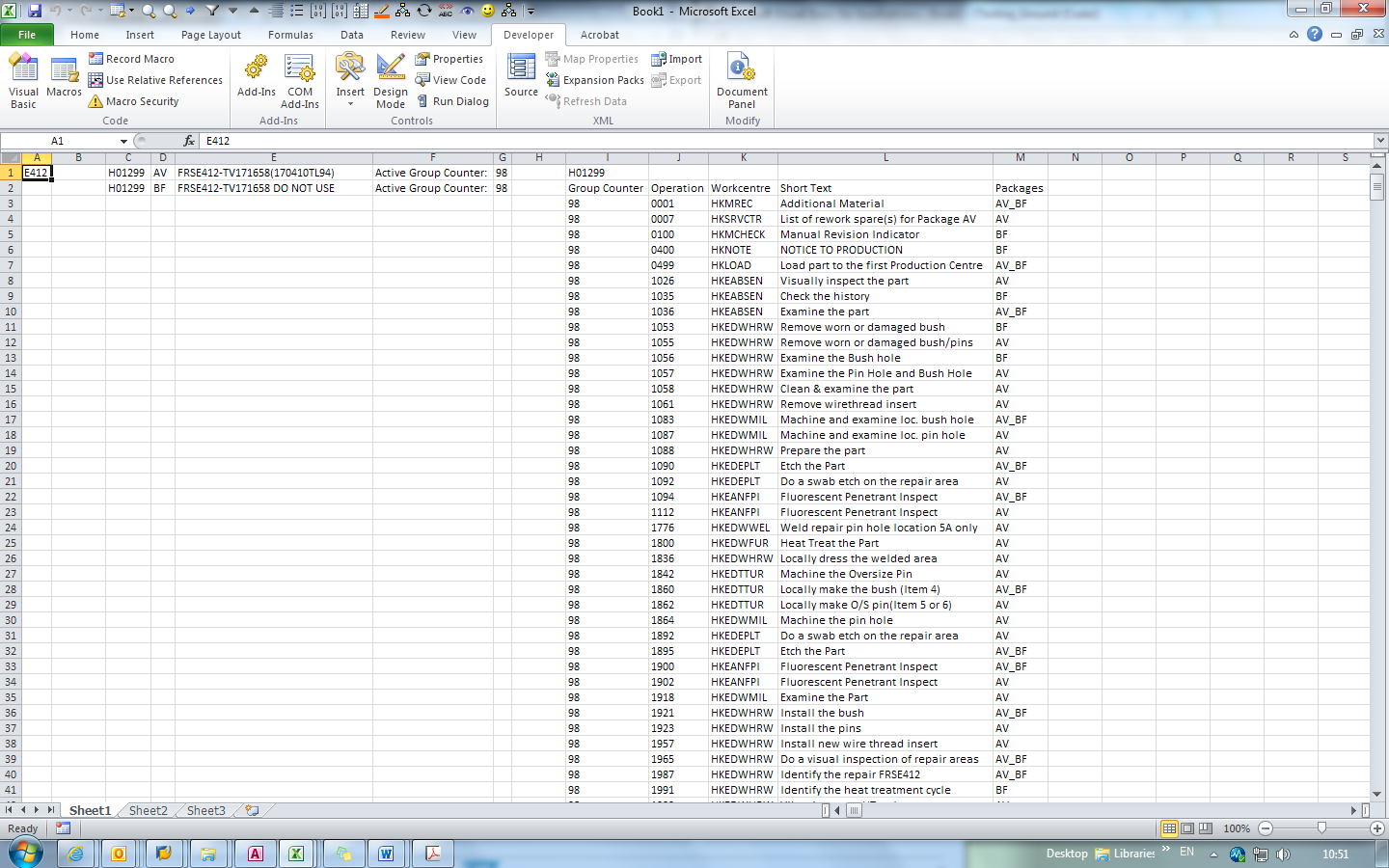
The following modules are depended by codes in other modules. They contain necessary codes and callable Marcos and functions for most of the codes to run properly. The major functionalities are listed as follows:

### SQ01: Operation finder

This is a set of Marco used to find details about a certain Repair/Plan Package, including its active group counter, operation numbers and work centre assigned.

This section has two callable Marcos:

* SAP\_Extract\_Ops\_From\_FRS() – Find by using package short texts, such as “FRSE412” or “TV167540”
* SAP\_Extract\_Ops\_From\_H0Plan() – Find by using Plan Number and Package. More than one package code can be added into the search criteria. (e.g. “H01203”, “AA”, “AB”)

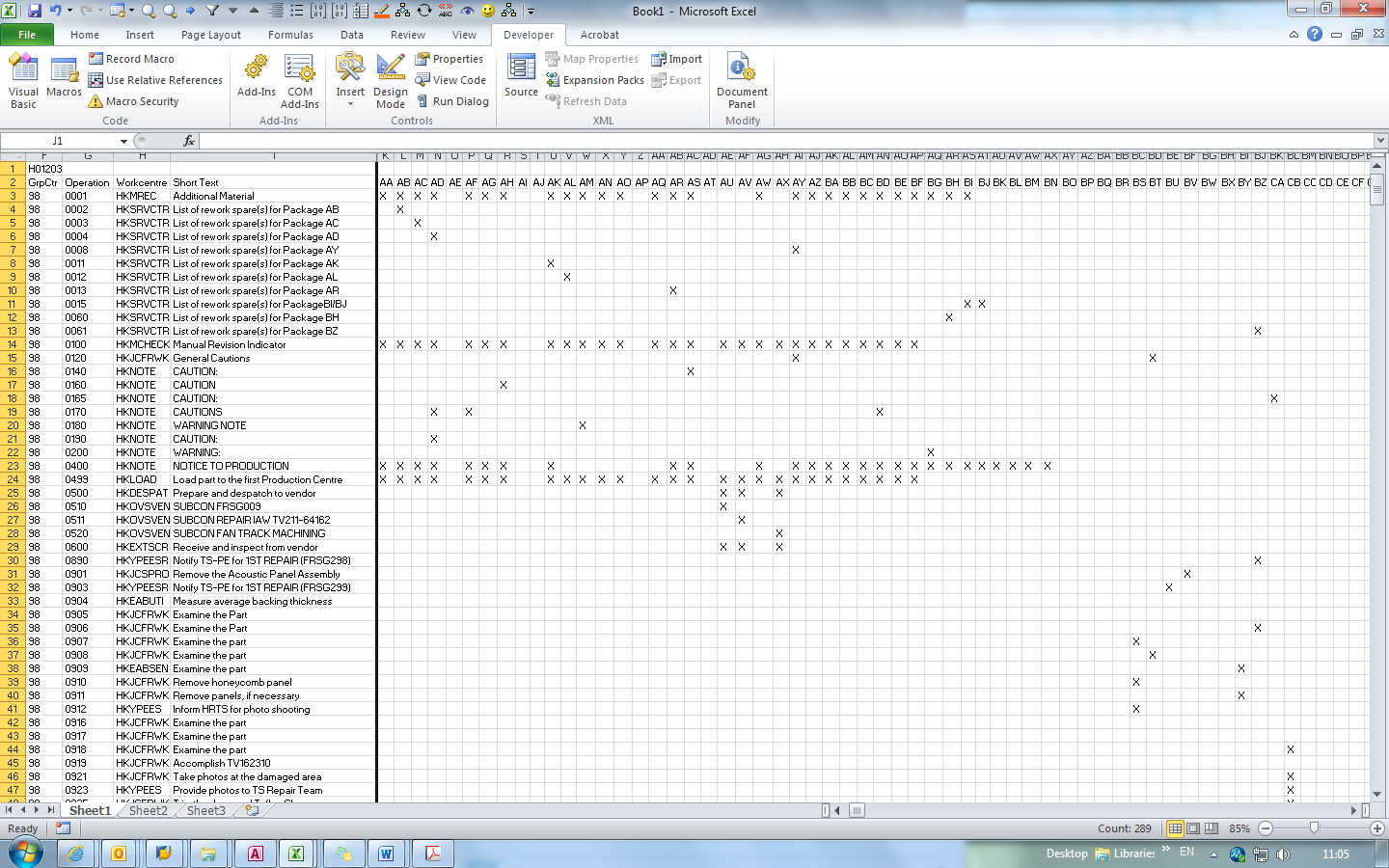
Fig.3.Typical instance after running SAP\_Extract\_Ops\_From\_FRS().

### SQ01/ZIP11/IP03: Plan details viewer

Marco of this section creates a very detailed sheet about a certain Plan Number in SAP, similar to that generated by SAP transaction “ZZA06”, creating a matrix showing relationships between operations and packages within the plan.

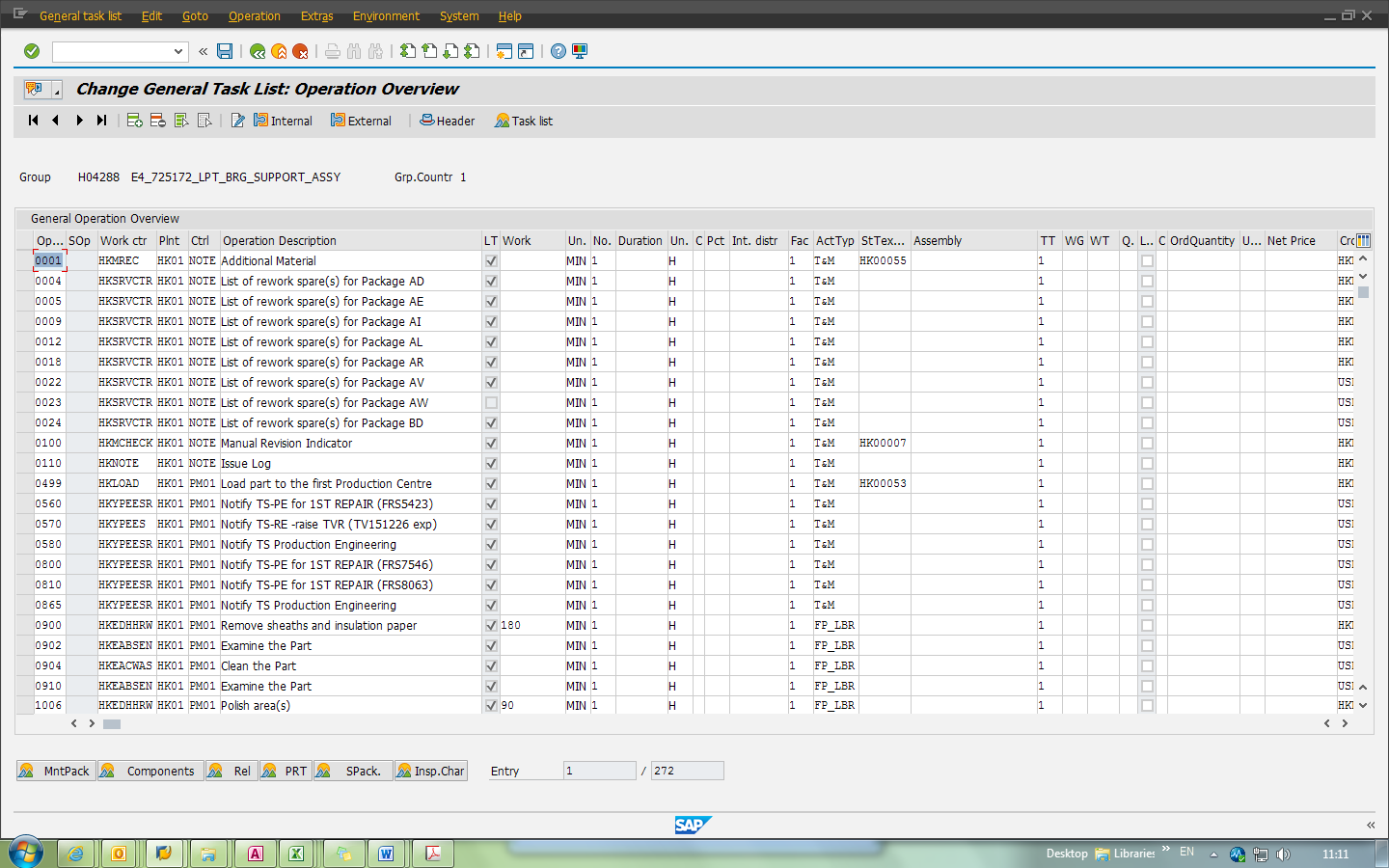
This section contains one callable Marco:

* SAP\_DetailOps() – Only one input variable: Plan Number.

Fig.4.Typical instance after running SAP\_DetailOps().

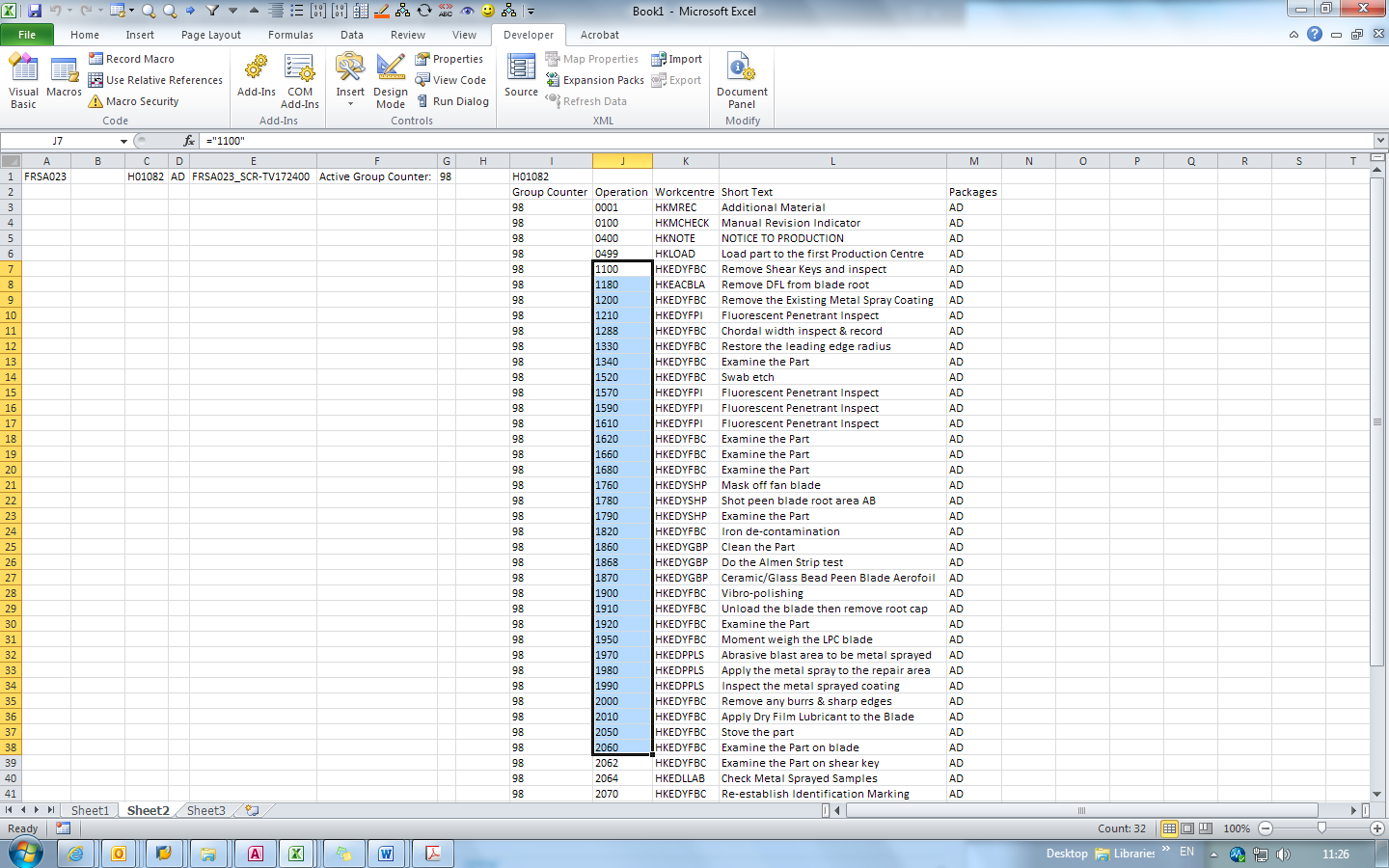
### IA06/IW32/ZL07: operations selector/un-selector

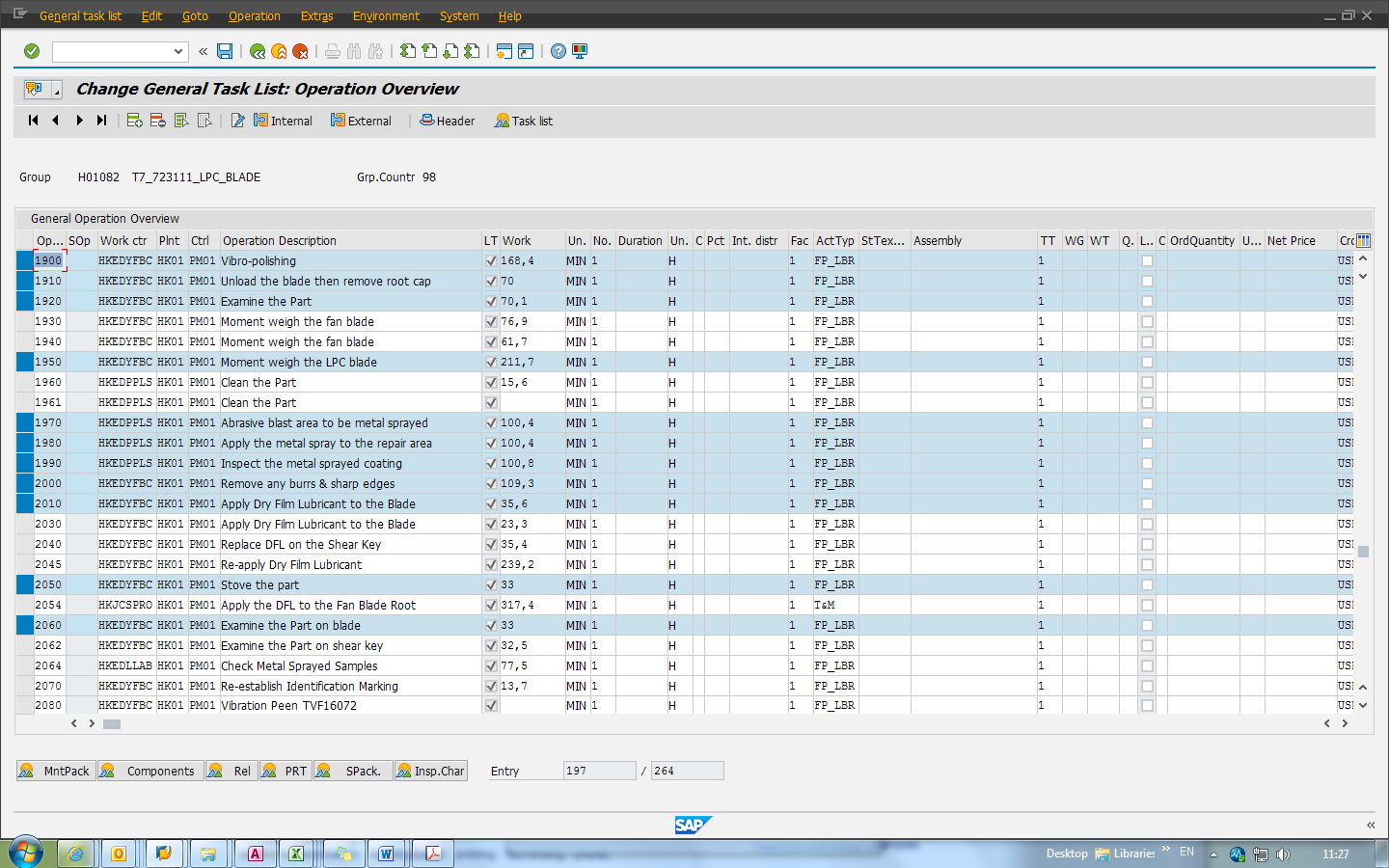
This set of Marcos is used to select SAP operations. The callable version of this selector takes in input using currently selected cells. This set of Marcos MUST be ran under “Operation Overview”:

Fig. 5. “Operation Overview” Screen. Applicable to IA06/IW32/ZL07.

This section contains two callable Marcos:

* SAP\_selectops\_callable() – Used to select operations in SAP based on the value contained in currently selected cells in Excel. Will only take in selection with all cells containing 4-digit numbers. Applicable to IA06/IW32/ZL07.
* SAP\_ia06\_UnSelectOps() – Used to select all operations in a SAP Plan/PMO EXCEPT for the values that match currently selected cells in Excel. Will only take in selection with all cells containing 4-digit numbers. Applicable to IA06 only.

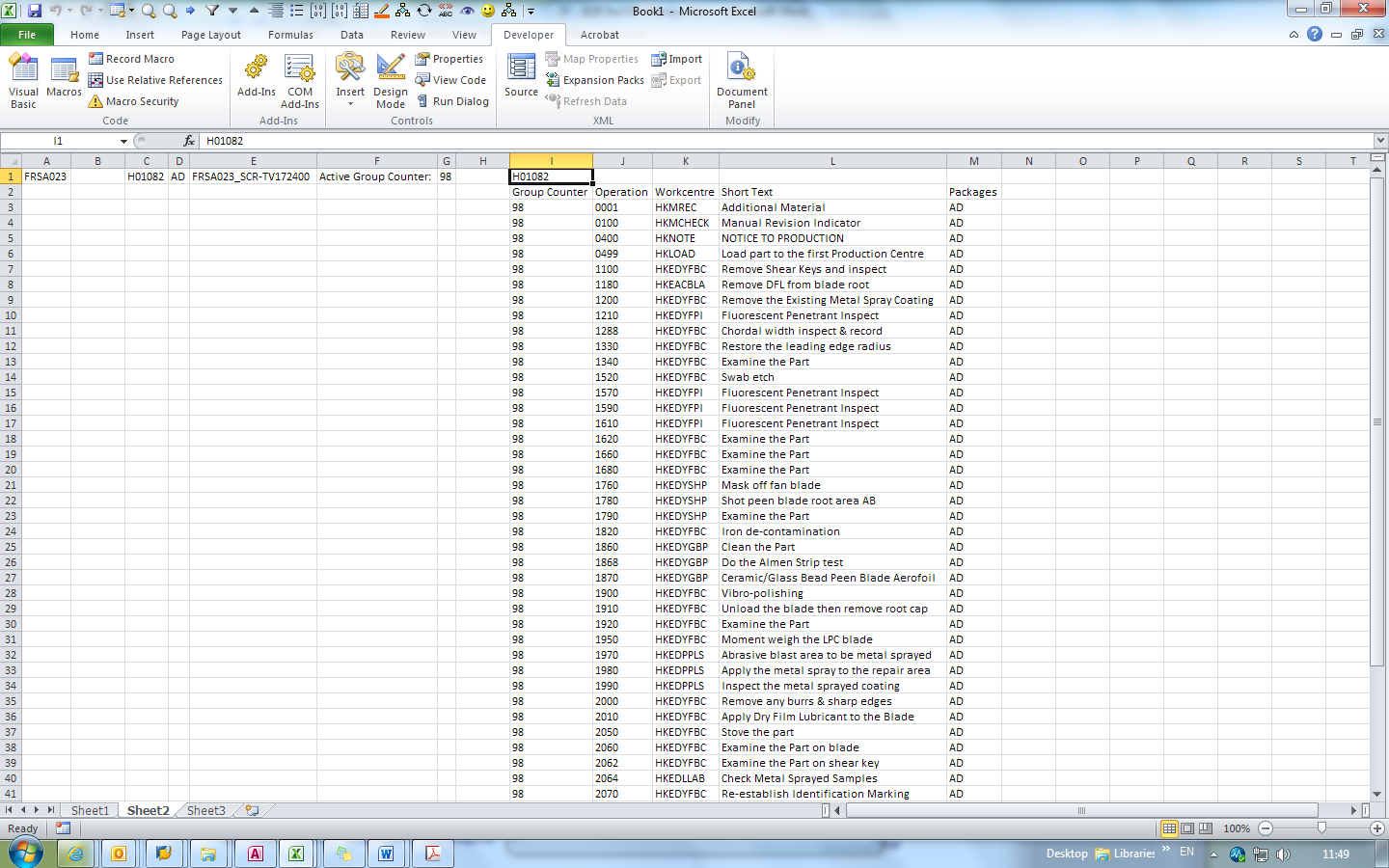


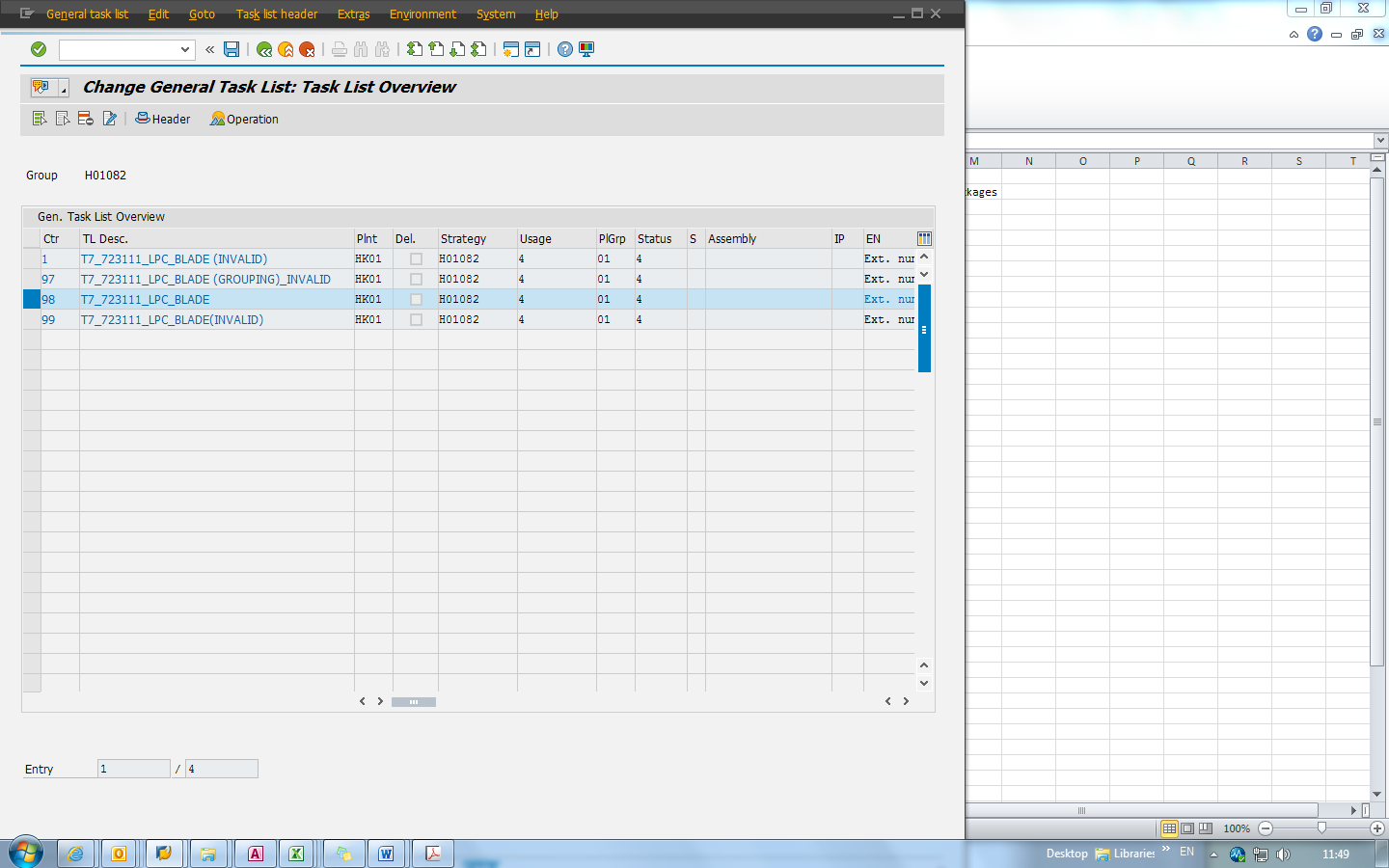
Fig. 6 & 7. Typical instance of selection and result, before and after running the Marco.

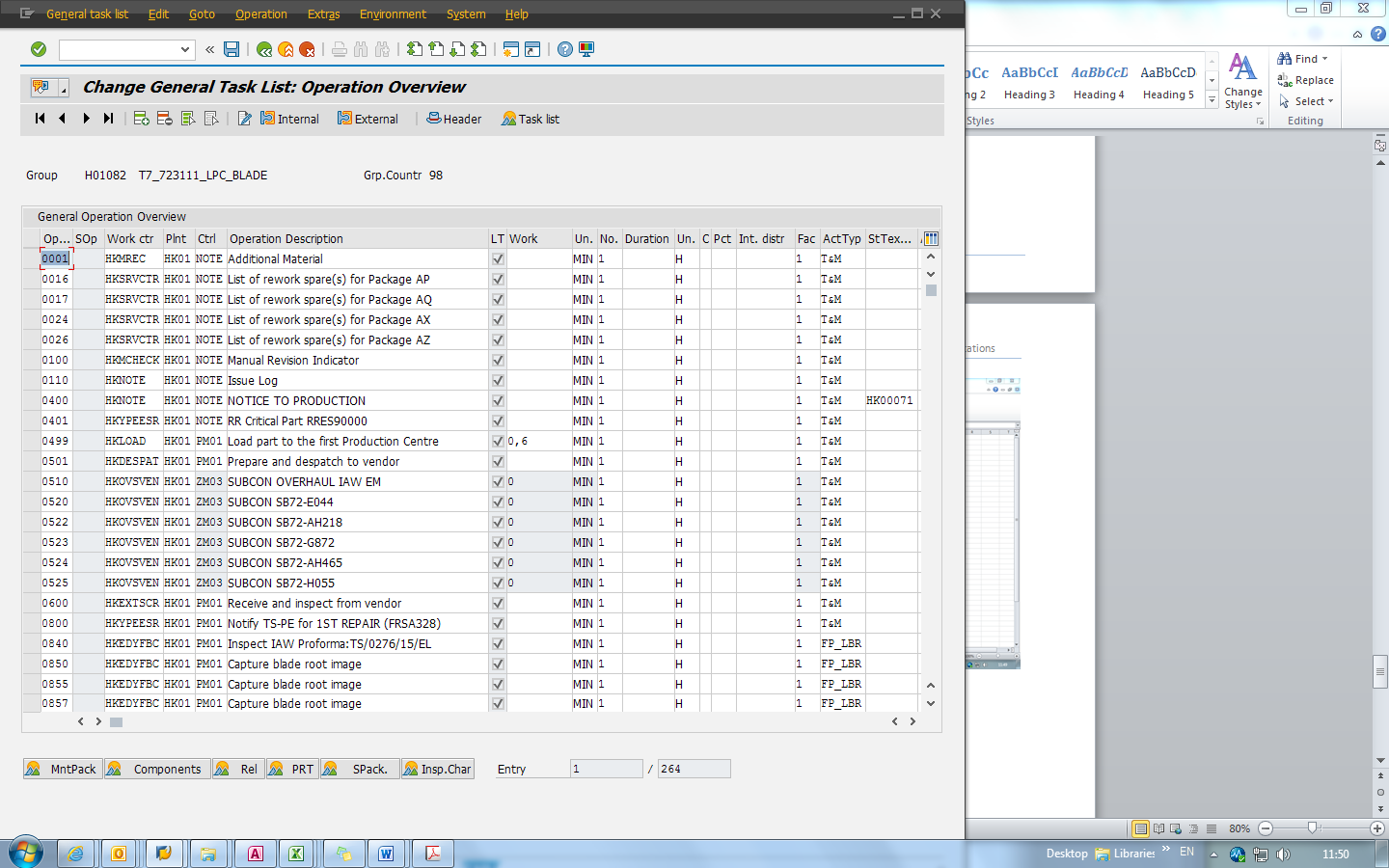
### IA06: GotoPlan

This Marco is used to quickly transverse into a plan editing screen of IA06 in SAP.

* SAP\_ia06\_GoToPlan() – Callable version. Will take in plan numbers by selection in Excel only and go into first non-“DO NOT USE” and non-“INVALID” named group counter.

Fig.8. SAP\_ia06\_GotoPlan() will only take in Plan Number in selection cell.

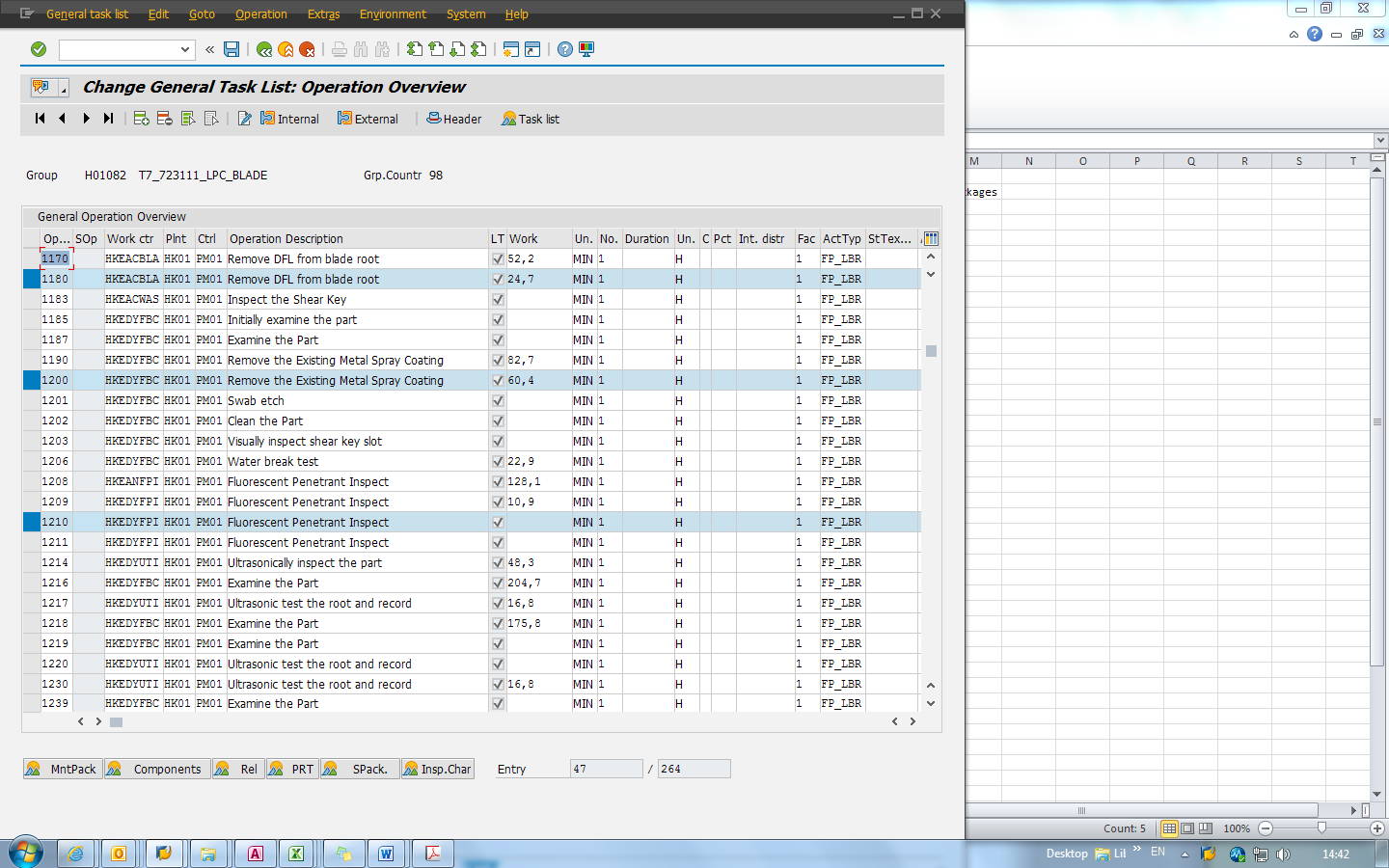


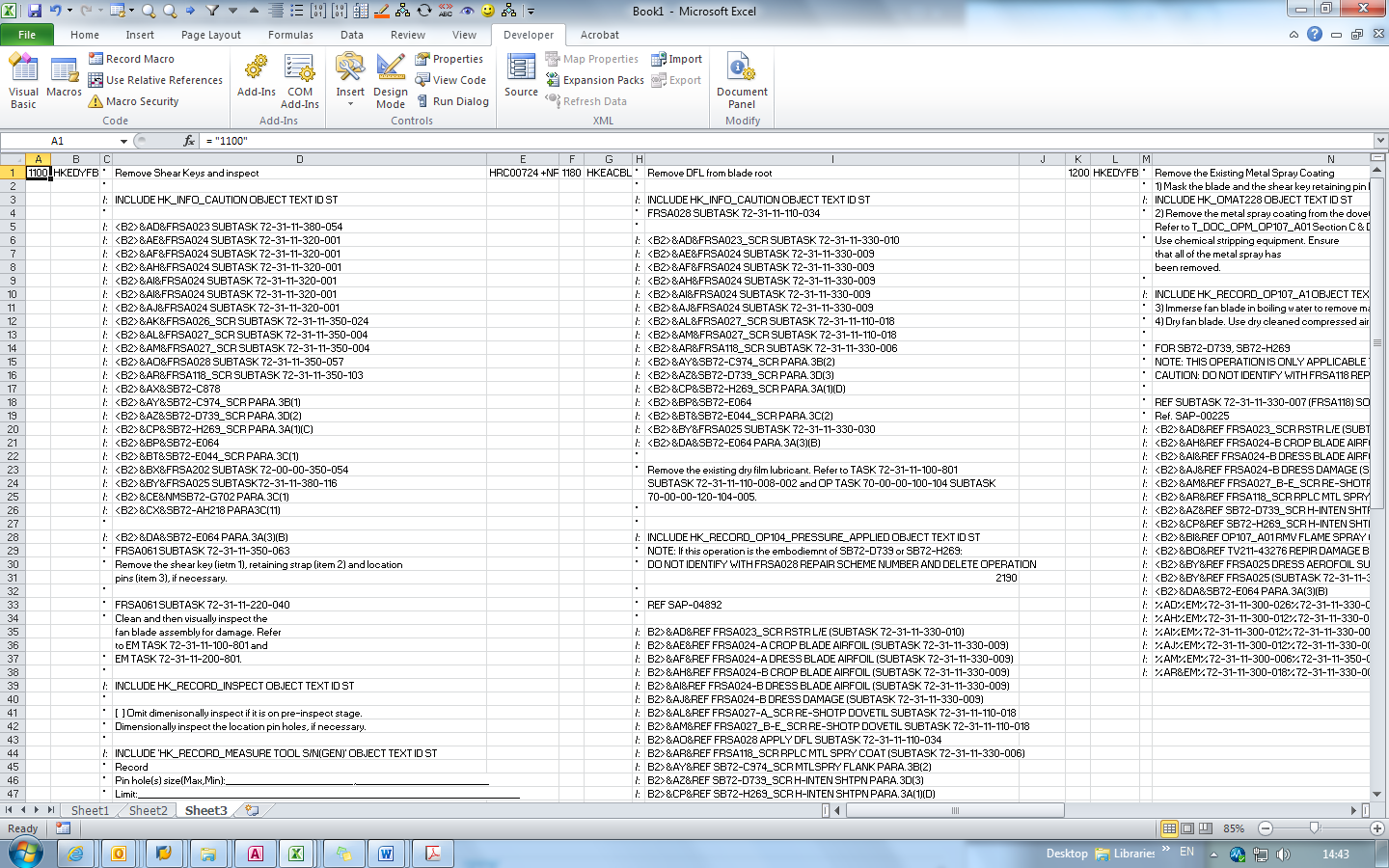
Fig. 9 & 10. Typical instance of running SAP\_ia06\_GotoPlan().

### IA06/IW32/ZL07: long text copier & paster

This set of macro is used to copy/paste SAP operation to/from Excel. Must be used in “Operation Overview” Screen (Sam as SAP\_SelectOps\_callable, refer to part d.).

* SAP\_longtext\_copier() – Copies the selected operations in SAP. Will overwrite cell contents in active worksheet. Applicable to IA06/IW32/ZL07 (not IA07)
* SAP\_longtext\_paster() – Copies the data in the active worksheet into the currently opened SAP “Operation Overview” page. Use the same format as SAP\_longtext\_copier() does. When used to a ZL07 session, the Marco WILL overwrite any operations sharing the SAME OPERATION NUMBER with the ones in excel. If used to a IA06 session, it will throw an error instead.

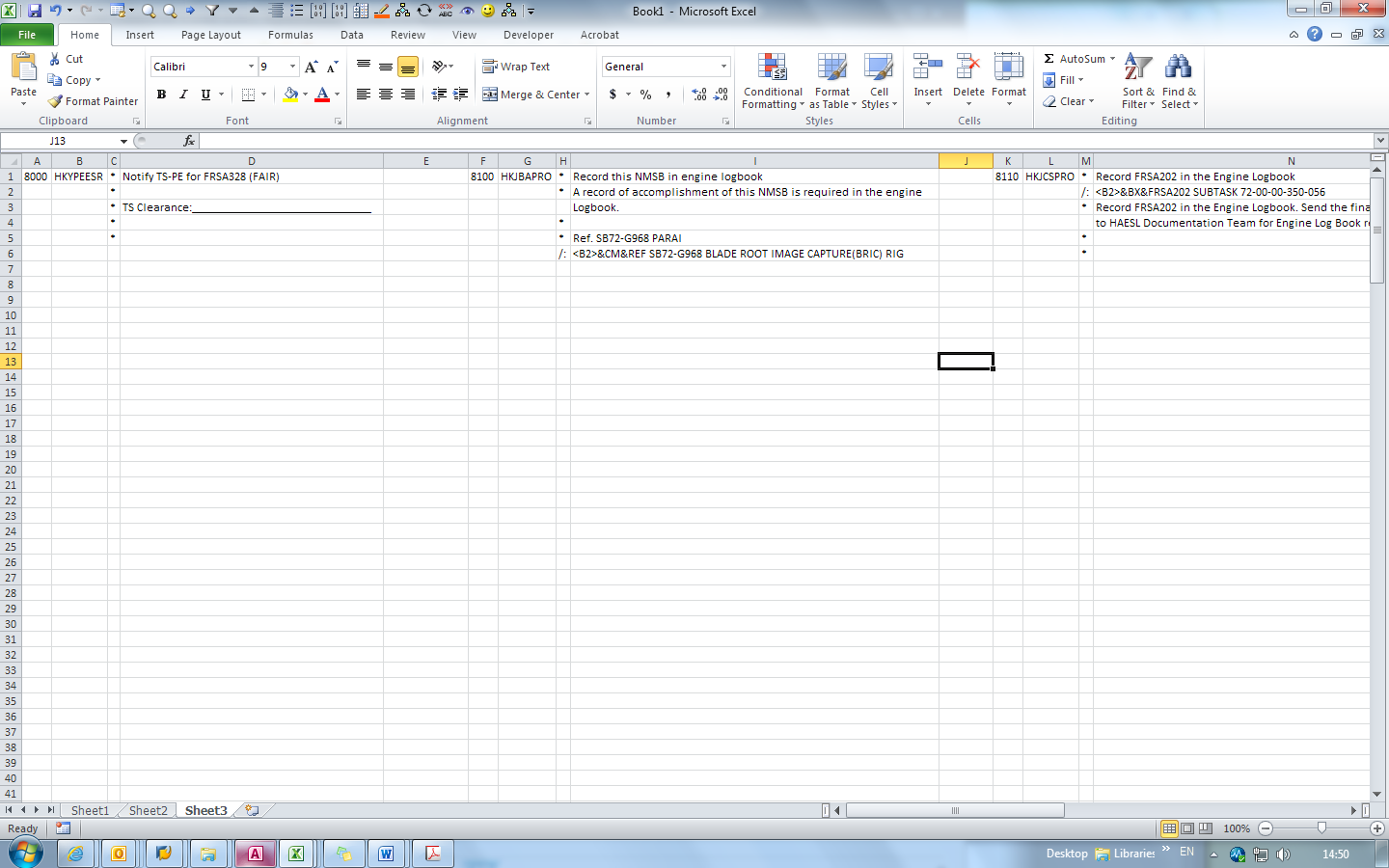


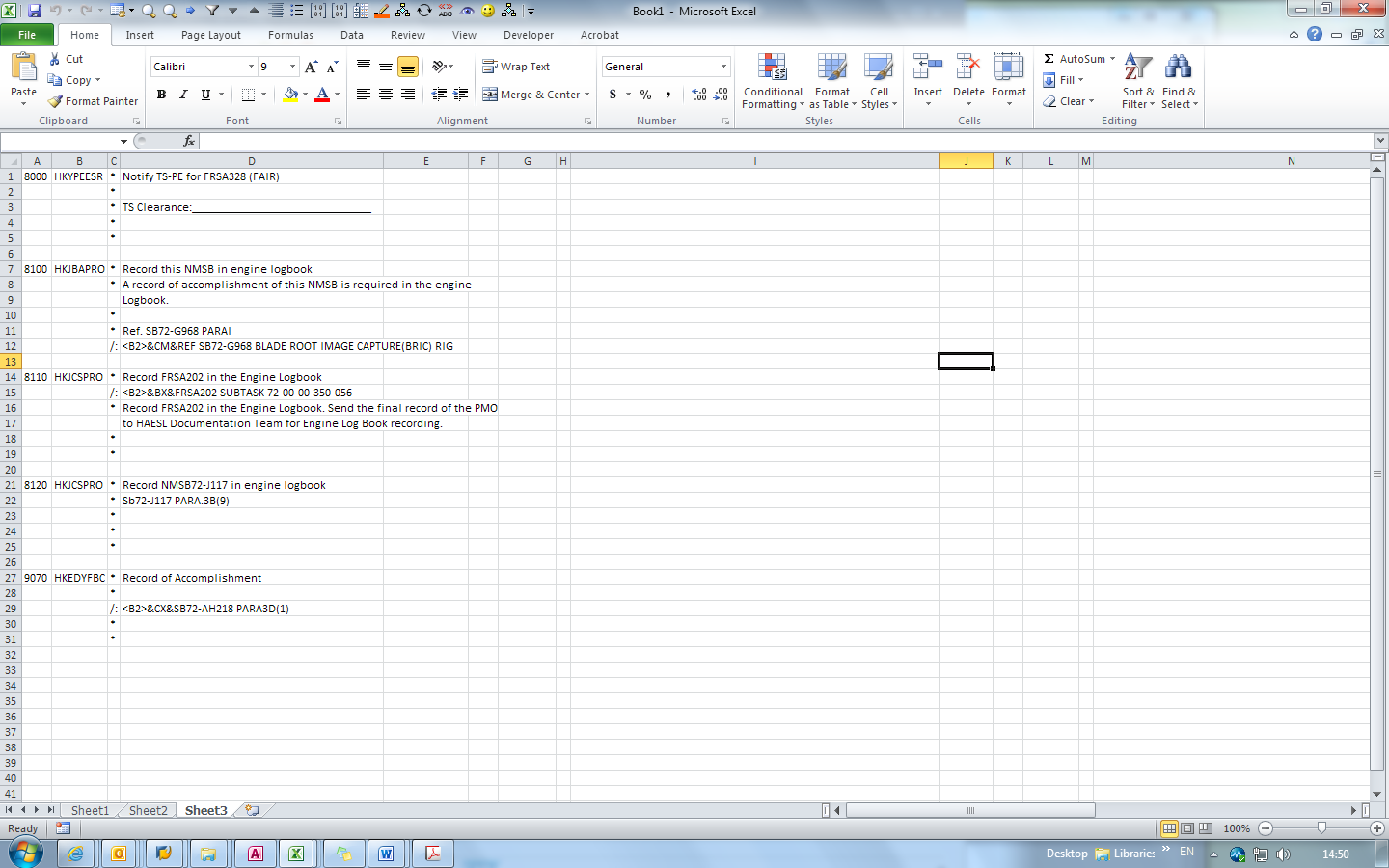
Fig. 11 & 12. Copying data of selected operations into the active sheet.

### Excel: Converting copied operation into vertical form

This Marco is used to convert the formatting of the copied operations in excel into a more readable format.

VerticalOps() – No input required. Will change the cells content in the active worksheet.



 Fig. 13 & 14. Converting the downloaded data into vertically displayed format.

### IA06: Long Text Replacer

This set of marco is used to replace texts in operations within SAP Session IA06. This will read selected operations in “Operation Overview” Screen in IA06 (Refer to Figure 7).

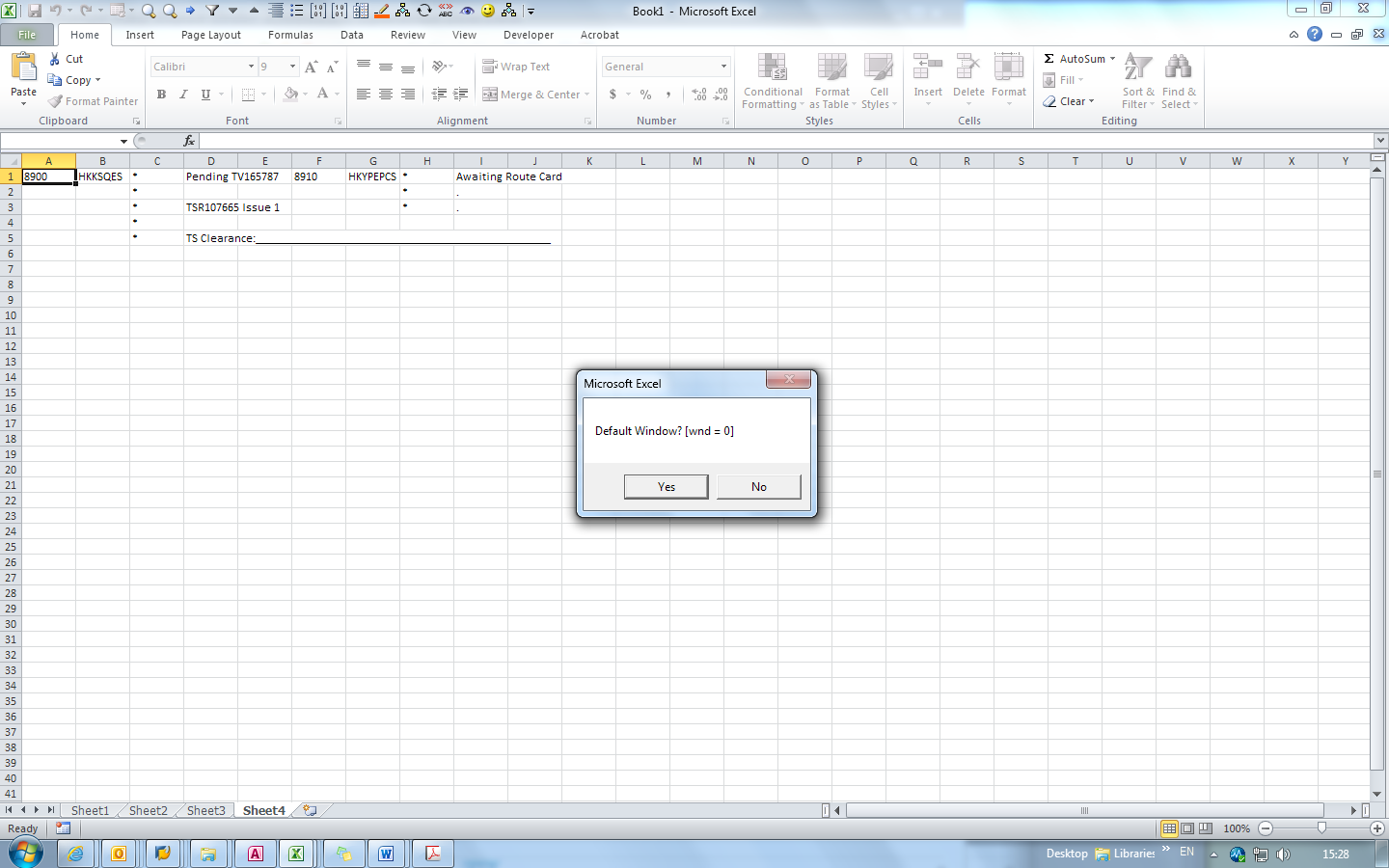
For replacing texts in ZL07, a callable sub zl07\_replace() is available but will not work except modifying the text to be searched and replaced in prior. Do not use if you have no experience in modifying VBA codes

* Operation\_Long\_Text\_Replacer\_callable() – For IA06. Replace any text matches first input with the second input.
* zl07\_replace() – Callable but not functional without tampering the codes.

### ZL07: Pending TV operations adder

This is a modification of the Marco SAP\_longtext\_paster() to make it feed in pre-set data, so that it will output two operations in ZL07: “Pending TVXXXXXX” and “Awaiting Route Card”. It will also prompt to print out the SPMO directly after adding the two operations. It will work on Session ZL07 with “Operation Overview” active (Refer to Figure 7).

SAP\_zl07\_PendingTV() – It will ask user to input the two operation numbers (for the said “pending TV” operations), TV number (e.g. TV154797) in waiting and the TSR reference(e.g. TSR106556 Issue 1).

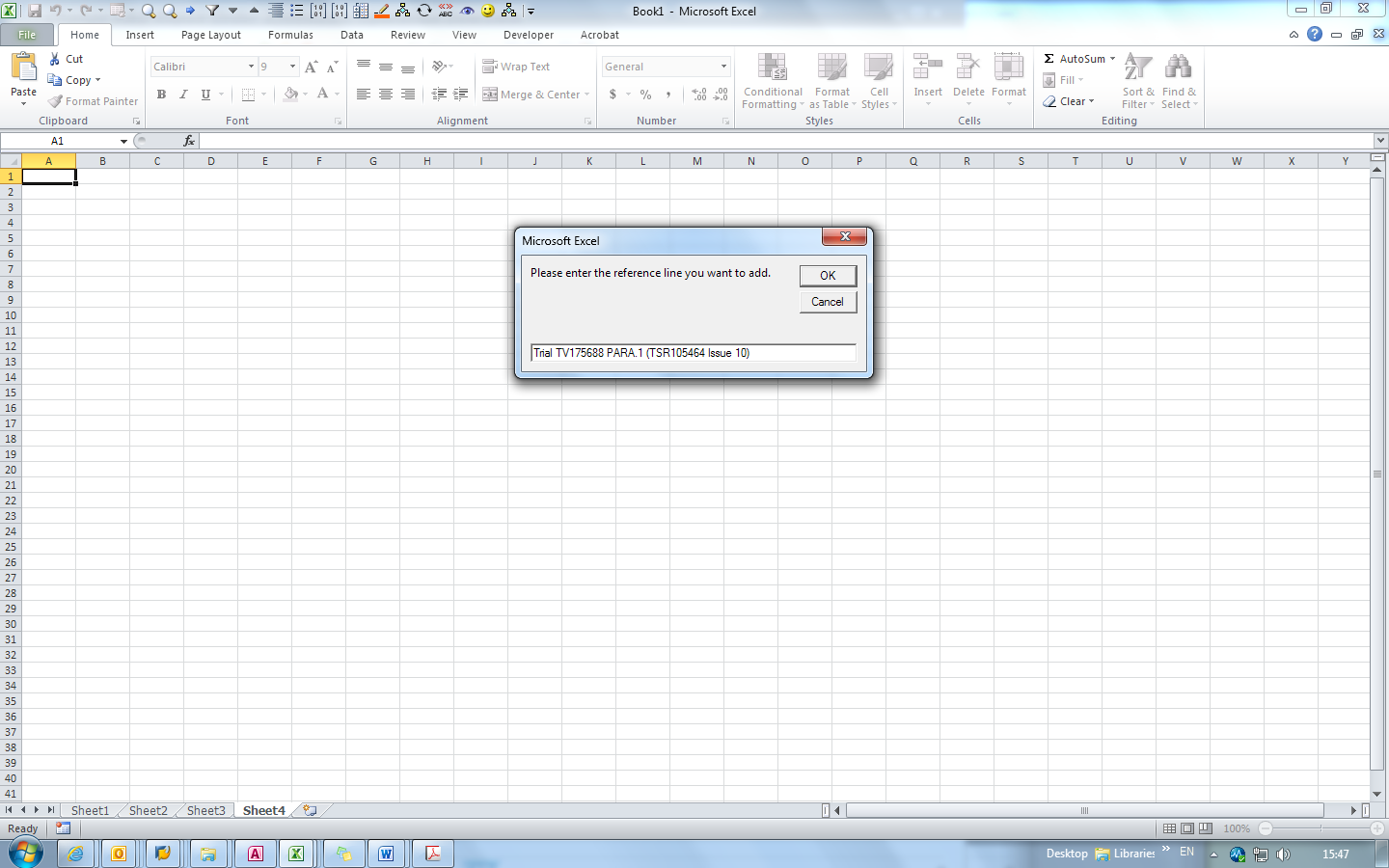
 Fig. 15. Pending TV operations created ready to be added into the SPMO.

### ZL07: Reference adder

This Marco is used to add references into selected operations in session ZL07. It will only work as the “Operation Overview” Screen (Refer to Figure 7).

SAP\_zl07\_AddRef() - The Marco will first ask for the reference line to be added. Then it controls SAP and goes into selected operations to add the entered reference line into appropriate position.

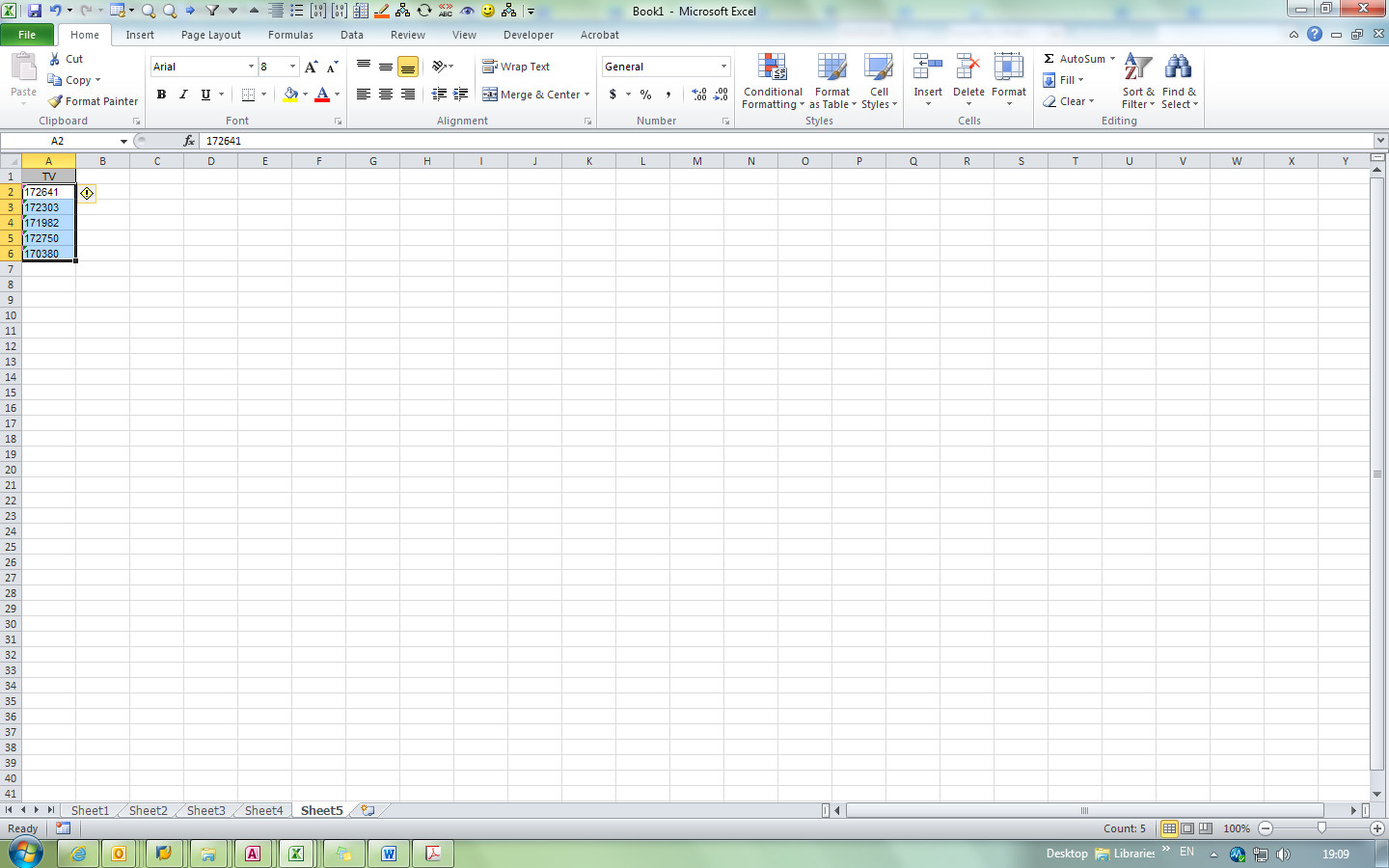
Note: It is not necessary to add “Ref.” before the actual reference as the Marco will differentiate Transformed and non-Transformed operations.

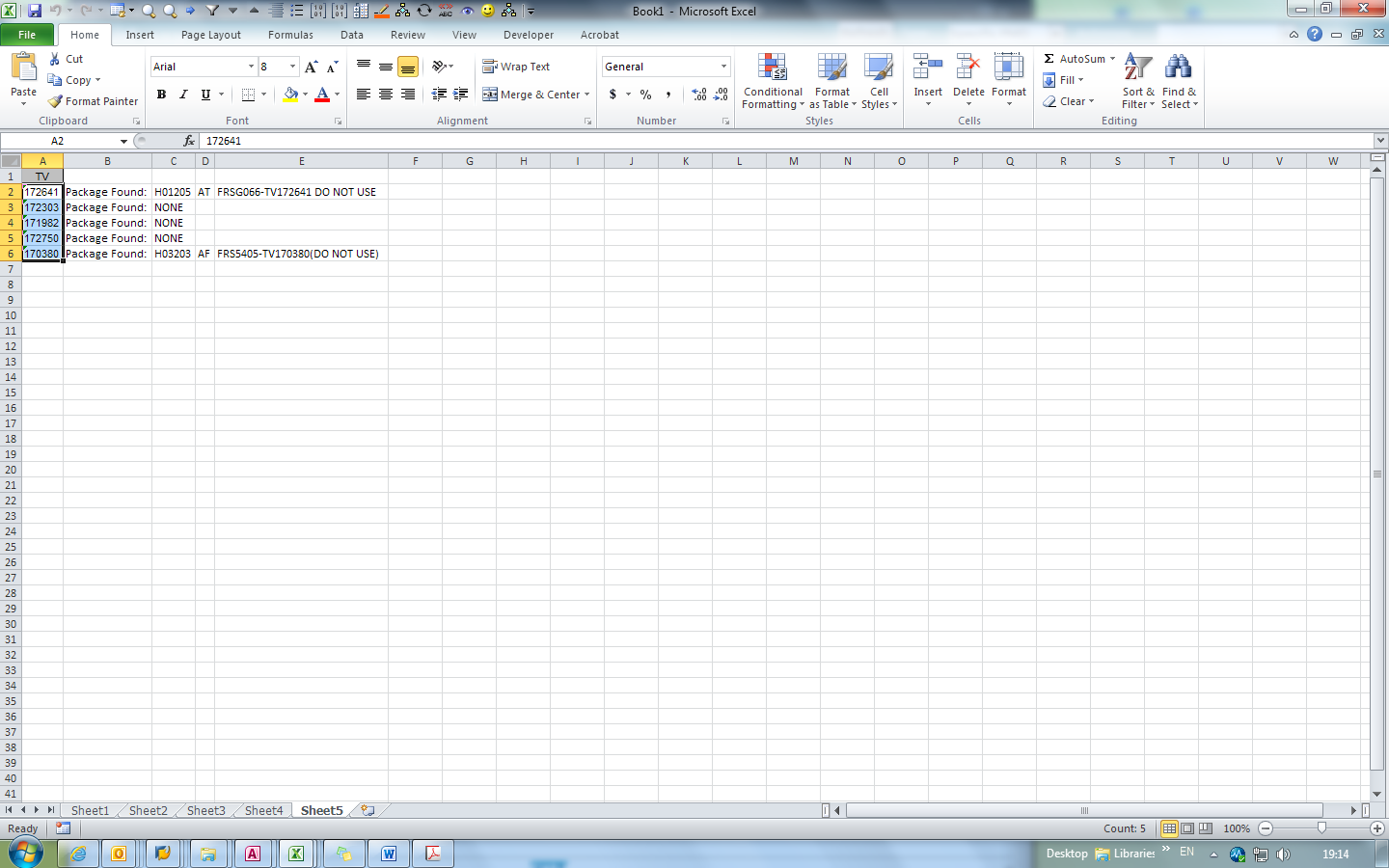
Fig. 16. It is not necessary to add “Ref.” before the actual reference as the Marco.

### SQ01: Simplified Plan Finder for Bulk Query

This Marco shares the same functionality as a plan number finder in section a. However, it is more of a simplified version designed for speedy checking.

SAP\_BulkChecking\_From\_FRS() – Take in currently selected cells as keywords and search them in SQ01 to see if there is any package short texts matches. If yes, the related plan numbers, package codes and package short texts will be recorded at right side of the selected cell.



Fig. 17 & 18. Example of using the Marco to bulk-check Repeater TV in H0 plans.

### Excel: Concatenating Strings

This Marco is used to concatenate data in several cells in succession into different formats.

* Concen() – Take in current selected cells and organize them into a table of rows which has a maximum of 72 characters each row. It is used to paste the data into SAP long text editor.
* Concen2() – Take in current selected cells and concatenate them into a single cell text string.

|  |  |  |
| --- | --- | --- |
| Original String set | Result of Concen() | Result of Concen2() |
| The manual content is to be revised in accordance with the aforementioned RMR. Pending formal | The manual content is to be revised in accordance with the | The manual content is to be revised in accordance with the aforementioned RMR. Pending formal Manual revision, this Repeater TV is issued to authorise Repair Organisations to work to the intent of RMR with immediate effect. This Repeater TV may be applied more than once until 10 March 2018. After 10 March 2018, the affected components or assemblies may continue in service until the next shop visit exposure at which time they should be re-assessed to the respective Manual requirement. |
| Manual revision, this Repeater TV is issued to authorise Repair Organisations to work to the intent of | aforementioned RMR. Pending formal Manual revision, this Repeater TV is |  |
| RMR with immediate effect. This Repeater TV may be applied more than once until 10 March 2018. | issued to authorise Repair Organisations to work to the intent of RMR |  |
| After 10 March 2018, the affected components or assemblies may continue in service until the next | with immediate effect. This Repeater TV may be applied more than once |  |
| shop visit exposure at which time they should be re-assessed to the respective Manual requirement. | until 10 March 2018. After 10 March 2018, the affected components or |  |
|  | assemblies may continue in service until the next shop visit exposure |  |
|  | at which time they should be re-assessed to the respective Manual |  |
|  | requirement. |  |

##### Table 1. Original text string and the result of execution of concen() and concen2().

### Excel: Comparing values in two columns

This Marco is used to compare data from two columns at the same spreadsheet.

* CompareRow\_callable() – Callable version, will ask for five inputs:
  + First Row (Master): Column to be highlighted. (e.g. Column “C” is to be highlighted, enter “3” as it is the third column of the spreadsheet.)
  + Second Row (Slave): Column to be searched. Will not be highlighted.
  + Color: The colour code for the cells to be highlighted. (e.g. 255 for the colour red, 65535 for yellow)
  + Exact match: If yes, the Marco will only highlight exact matches. If no, any wording matching the keywords from the Slave supplied by the Master will suffice in triggering a highlight action.
  + Sorted match: Press yes if both rows are properly sorted. Will speed up searching. Press no otherwise.

### Excel: Extracting Rows with certain value

This set of Marco is used to extract rows in spreadsheets with specified value. Other rows without the said value will be deleted.

* ExtractSUBTASK() – Will ask for which column to search and delete rows without SUBTASKs with in.
* Extract\_String\_of\_certain\_Length() – Will ask for the keyword and the length of the string that needs to be extracted.

### Excel: Highlighting rows

This set of Marco is used to highlight cells in spreadsheet in a specified column.

* HighlightOPs\_callable() – Callable version of highlightops(). Will ask for three inputs:
  + Col: Column to be searched
  + String: Keyword to be searched
  + Color: The colour code to highlight the hits. Refer to section l.

### IA06: Transformation-related Marcos

For the set of Marcos handling the Transformation-related procedures, a separate pack along with its own SOP has been published individually.

Therefore, their functionalities and usage will not be discussed in this document. Please refer to their own SOP at the following link:

[V:\Technical Services\5. Task List Planning\5.2 Team Projects\SAP Marcos\Transformation\_SAP\_Marco\_Pack\_for\_Excel-26OCT15\Readme-26OCT15.docx](file:///V:\Technical%20Services\5.%20Task%20List%20Planning\5.2%20Team%20Projects\SAP%20Marcos\Transformation_SAP_Marco_Pack_for_Excel-26OCT15\Readme-26OCT15.docx)

## Marcos in “Auxiliary/Supplementary” Modules

### Generating Snapshot & Search Function

This set of Marcos Modules, “Snapshot\_dump” & “The\_Grand\_Search”, has been published individually under the module names “Snapshot\_Dump\_Standalone” & “Snapshot\_Search\_Standalone”. The standalone version Marcos are the same as the non-standalone version.

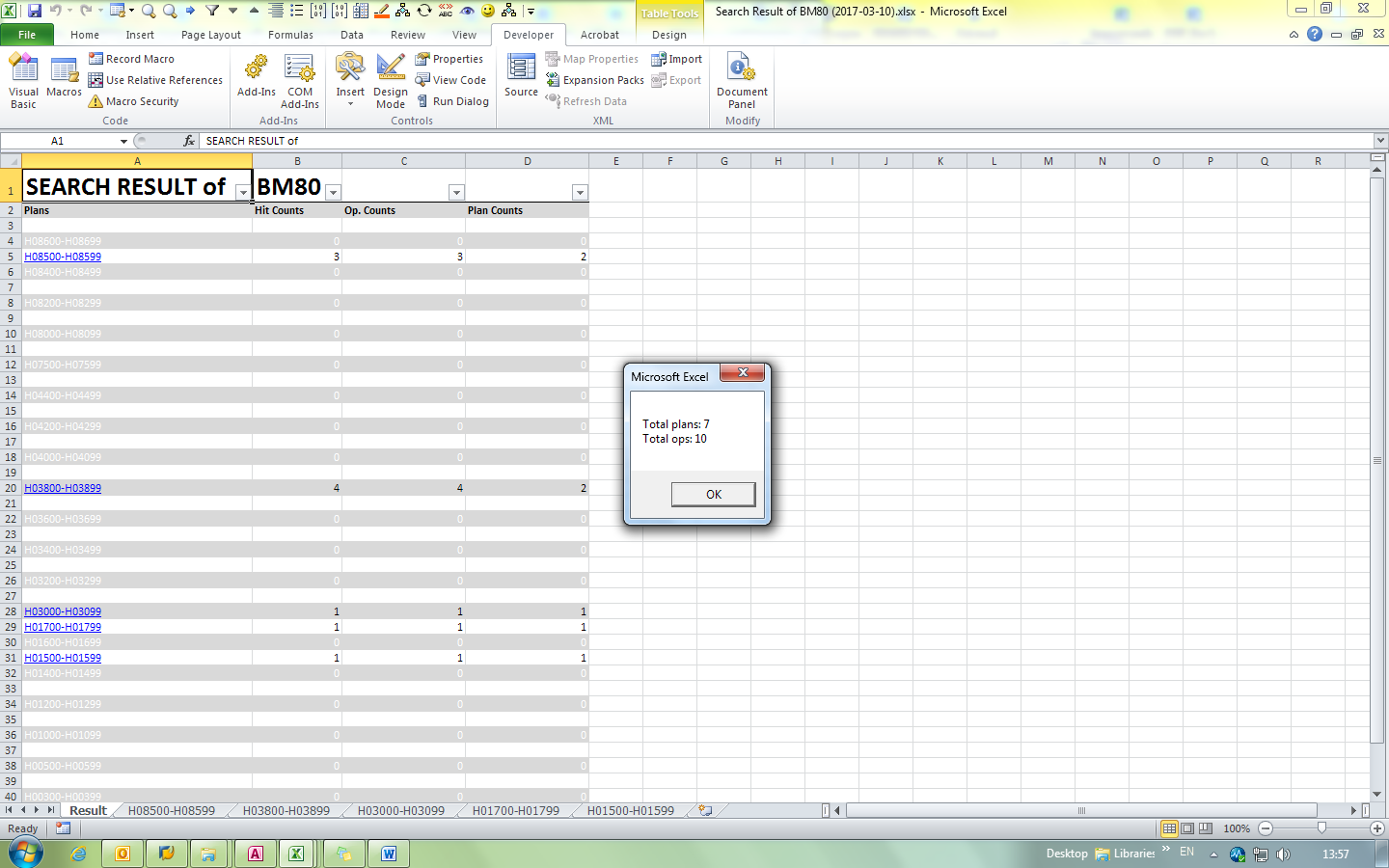
Therefore, their functionalities and usage will not be discussed in this document. Please refer to their own SOP at the following link:

[V:\Technical Services\5. Task List Planning\5.2 Team Projects\Snapshots\SOP for Generation of SAP Data Snapshots.docx](file:///V:\Technical%20Services\5.%20Task%20List%20Planning\5.2%20Team%20Projects\Snapshots\SOP%20for%20Generation%20of%20SAP%20Data%20Snapshots.docx)

### Hits, operations and plans counting for the search result table

This set of Marcos is used for counting-related purposes. It must be used with the search result table as active worksheet.

* CountingModule.main() – No input required. Will return operation and plan hit counts in the active worksheet.
* CountingModule.Recount() – No input required. Will re-generate the counts in “Result” worksheet.

Fig.19. Typical instance after running CountingModule.main().

### Comparing two search results

This Marco will compare two generated search result tables.

* ComparingModule.main() – Will ask for two inputs. The first one (Master) is the search result table which will be highlighted. The second one (Slave) will only serve as datatable.

### IA17: Counting keywords

This Marco counts keywords from SAP session IA07, utilizing the search function available in SAP GUI.

* CountTV() – Take the currently selected cells as the list of keywords to be searched. Will return a list of hit counts at the right of the selected cells. Must be used with IA17 screen is showing data.

### ZL07: Printing out WASH/INSPECT PACKs in plans

This Marco will print out the WASH/INSPECT PACKs in plans.

* zl07print\_WASH\_NDT.main() – Take in the current selected cells as the list of plan numbers. Will ask which functional location to print those PACKs out and offer choice to print just WASH PACK, just INSPECT PACK or both. Will delete the called out order once printed by default.

## Marcos in “Experimental/Outdated/One-off” Modules

Modules in this category are generally unusable without modifying the codes and thus this SOP will not discuss the Marcos within and their purposes. Please refer to the in-code comments for a more technical-oriented guidance.