MACROS USED IN DEVELOPMENT

CONTENTS

1. GENERAL INFORMATION

The Poly System may be set up either as independent Poly units or as a system with all the Poly units connected to a shared 200mm floppy disk.

1.1. STANDALONE MODE

In the Standalone Mode, the Poly is not connected with other units or any disk unit. Programs can only be entered through the keyboard, and no facility is currently available for saving them. Also only a restricted Polybasic is available. This contains most of the commands available in the full Polybasic, but with all disk handling and a few other commands omitted. In describing the commands, those not available in the Standalone mode are clearly indicated.

1.2. SYSTEM MODE

In System Mode, each of the Poly units are connected via a communications line to a central disk unit, on which all programs are stored.

1.3. THE POLY KEYBOARD

The Poly keyboard has been specially designed to handle the Poly features. The lefthand part contains a standard "QUERTY" keyboard. Special keys have been added on the other side. (See page 4).

Special keys include:-

Cursor control keys. These are used to move the cursor around the screen. They may be used when entering programs or data, but the back-arrow (backspace) key is that only key at the moment that is functionally usefull when entering a Basic program.

8 September 1981
File PPOCSY

Please note the following additions, and alterations to MACROS:

GSECTOR: For creating a line desciption of a full or part circle.

INTEXT: This inputs a string of a length defined by the user. Does not input any control characters.

INDIGIT: This inputs a numeric value only, X number of digits long as defined by the user. Also allows decimal point if parameter 5 equals 1.

MSKEL: Note that the parameters of this Macro have changed considerably.

3.1. PRINT@ (row, column), string

New command

This command references the current teletext screen and prints the string at the specified row/column. The cursor is set following the command, as for PRINT. The comma following the brackets is optional.

Example: PRINT@(12,34),A$;BB;

3.2. RND(X)

Changed function

If X > 1, a random number from 0 to X-1 is returned. A number less than 0 resets to a new random sequence. However this must not be a power of 2, i.e. do not use RND(-0), RND(-1), RND(-2), RND(-4), etc. This is a bug in the BASIC interpreter.

3.3. USR (address, parameter 1, parameter 2, parameter 3).

Changed function

The USR function calls a machine language subroutine. The address is the entry point in the subroutine. The 3 parameters are handed over in the D, X and Y registers respectively. If an error occurs, the value is handed back to the USR command in register A with the condition carry

flag set. If an error is encountered, then USR calls the BASIC error routine with the value returned. If a parameter is not set, USR hands over the value FFFF (-1). The parameter may take any integer values, negative or positive.

3.4. SPC(I%)

Changed function

The SPC function has been extended so that it may be used at any time to create a string of I% spaces. In this manner it can also create a string of I% Ascii characters by SPC(I%,n), where 'n' is a valid Ascii value. Either of the above methods may be used within a PRINT statement.

3.5. INCH$

Function to be changed

INCH$ at present returns a null string if nothing is waiting, or the character of a key that has been pressed. This function is going to be changed so that a channel parameter is given, as in the original version of Basic. i.e. INCH$(0) for the keyboard. Alternative to INCH$, are USR(0) and USR(1). USR(0) scans the keyboard to see if a key has been pressed. To get a valid value you must use USR(0) by anding it with 256 e.g. KE=USR(0) AND 256. This will return a 0 if no key has been pressed, and the value 256 if a key has been pressed. USR(1) acts in much the same way as INCH$, except that it waits for a key to be pressed. It returns the Ascii decimal value multiplied by 256 of the key pressed. It should therefore be used in a statement such as

```
A% = USR(1) / 256
```

to get a decimal value, and in a statement such as

```
A$ = CHR$(USR(1) / 256)
```

to get the character value.

3.6. Set Memory Limit USR(40969,parameter)

A USR callable function is available to set reset the upper limit of the BASIC area. To use this call USR(-24567, upper memory limit)

3.7. INPUT statement

Pressing ENTER and nothing else in response to an INPUT statement, leaves the value in the variable unchanged.

4.1. BEEP

Function:

Beeps the speaker. (Does not contain a †)

Parameters:

1. The number of beeps to be performed. Default 1.

4.2. BREAK

Function:

Enables and disables the BREAK key. (Does not contain a †)

Parameters:

1. ON or OFF. (Default OFF)

4.3. CALCULATOR

Function:

Enters calculator mode. This uses line 23 and returns to the program on the pressing of ENTER. This allows the user to do calculations during a program without affecting the program. (Not implemented yet)

Parameters:

none.

4.4. CLEAR23

Function:

Clears line 23 to spaces, and prints the screen number on line 23.(Does not contain a †)

Parameters:

Any parameter will cause it not to print the screen number.

4.5. CLS

Function:

Clears the current teletext screen to spaces. (Does not contain a †)

Parameters:

none.

4.6. COLOUR

Function:

This function sets the colour for the Background, G1, G2 and T1 80/20 screens. This colour is then used for all GPRINT function calls on the G1 and G2 screens and on the PRINT calls to the Teletext 1 screen in either 80 or 20 character mode. (Note both the T1(80) and T1(20) screens do not recognize inserted colour control characters. The whole screen must be set to one colour with no background on that screen). (Does not contain a †)

Parameters:

1. The colour. For the Background, this may be either:
   RED, GREEN, YELLOW, BLUE, MAGENTA, CYAN, WHITE or BLACK. The first character of each colour, except BLACK, may be used to indicate the colour. BLK may be used for BLACK.

For G1, G2, T1(80) and T1(20), the colours may only be:
RED, GREEN, BLUE, WHITE. Alternatively the first character of each colour may be used.

2. The screen. If this parameter is not given, the colour is set for writing to the current graphics screen. Otherwise
   B is entered for the Background or
   T is entered for the T1(80) and T1(20) screens.

4.7. DISPLAY

Function:

This controls the screens which are displayed. (If T2 or G2 are specified, the code contains a †. The code for T1 and G1 does not contain a †)

Parameters:

1. Screen type. This may be either:
   T1 for teletext 1
   G1 for graphics 1
   T2 for teletext 2
or G2 for graphics 2.
The priority is fixed as above. Screens G1, T2 and G2 may be mixed using the MIX macro. The background is always displayed. To switch the background off, set it to BLACK using the COLOUR macro.

2.ON or OFF. Specifies whether the screen display is to

be turned ON or OFF. If this parameter is not specified, the display of the screen is switched ON.

4.8. EDITOR

Function: Sets up a defined area on the screen allowing the user to enter data on the screen within that area. Full cursor control is available. The area is not cleared prior to EDITOR. The user exits from the EDITOR routine by pressing ENTER. The user may insert his own edit on the keys that may be pressed. Following EDITOR, the text may be moved from the screen by TEXTGET or be automatically placed into the array specified in parameters 6 and 7. (The code contains a †).

Parameters:

1. Start row of edit area.
2. Start column of edit area.
3. End row of edit area.
4. End column of edit area.
5. Name of users own routine. This routine is given the current key pressed in Z%. The programmer may insert his own edits on the key value and if an invalid key has been pressed, change Z% to 0 before the RETURN. The routine should be placed in position SUB.
6. The number whose multiple will leave a blank column. If not entered then no blank columns will be left.
7. Name of the string array into which the screen data is to be placed after ENTER is pressed. Each row on the screen is placed in a separate occurance.
8. Value of first element in the array in which the data on the screen is to be placed - default 0.
9. GET if the screen is to be loaded with the values in the array. PUT if the values on the screen are to be loaded back into the array. Both of these only apply if parameter 7 is entered. If parameter 7 is entered and no value is put into parameter 9, then both GET and PUT are done. If a virtual array is specified, it is up to the programmer to allocate, open and close this.

4.9. GCIRCLE

Function:

Draws a circle of given radius and specified centre on the current graphics screen in the current colour. (Does not contain a †).

Parameters:

1. Pixel row of centre.
2. Pixel column of centre.
3. Horizontal radius in pixels.

4.10. GCLS

Function:

Clears the current graphics screen to spaces. (Does not contain a †)

Parameters:

None.

4.11. GCONVERT

Function:

Converts a line description to a boundary and places it in the last defined boundary area. Also, this will store in a defined string, the current boundary stored in the LDB area. (Does not contain a †)

Parameters:

1. Name of string containing the line description. If this parameter is not given, then the conversion to a

boundary is not done. The conversion uses the work screen area, that is the virtual screen area if one has been defined, and the non current screen area otherwise.

2. The name of the string into which the boundary description is to be placed. If not given, then the boundary is not stored in a string but just left in the LDB area.

4.12. GETNAME

Function:

Gets the users Christian and surnames. These are each strings of up to 12 characters. (Does not contain a †)

Parameters:

1. The variable name of the string into which the christian name is to be placed.

2. The variable name of the string to which the surname is to be placed.

Only one of the two parameters is necessary.

4.13. GFILL

Function:

This fills in the area within the defined boundary on the current graphics screen with the current colour. The pattern of pixels may be defined. This function converts a line description as a boundary. (see GCONVERT.) (Contains a † only when any of parameters 2, 3 or 4 are entered as negative values, or as a variable or expression)

Parameters:

1. The string variable which contains the boundary description. If the string contains an invalid boundary description an error is returned. If no string is given, the whole screen is cleared. If LDB is specified then the Last Defined Boundary is used.
If the value LINE is entered in this parameter, then the GLINE parameters may be specified as in GPRINT, the line description will be converted to a LDB and the area filled with solid colour (Pattern 63). The line string is not kept.

2. Pixel row for the placement of the start of the boundary. Even though a boundary description places it at a particular place on the screen, the programmer may specify new initia coordinates for its placement. If parameters 2 and 3 are not given, the boundary is taken as definded in the string.

3. Pixel column.

4. Pixel pattern. The programmer defines which pixels are to be displayed in each group of 6 pixels in a row, over the whole filled area. The pixels are represented as a 6 digit binary number with 1 representing ON and 0 for OFF. Parameter 4 the value of the 6 pixels. This can be easily calculated by assigning the value 32 to the first pixel, 16 to the second and so on. A pixel pattern of 101101 is entered as 45 (32 + 8 + 4 + 1). If no pixel pattern is given, all pixels are turned OFF within the boundary, i.e. the area is cleared. To CLEAR the current graphics screen, the programmer need only enter:

```
M GFILL
```

5. Shift number. During a FILL, each row of pixels may be shifted up to 5 positions to the right. A Shift of 0 gives vertical lines, a shift of 1 gives right diagonals, etc. If not specified, a shift of 1 is made.

4.14. GLINE

Function:

GLINE sets up a line description for drawing a line or defining a boundary on the graphics screens. The line may be displayed using GPRINT. If the line description set up has left and right boundaries defined it can be used as a boundary in GSWAP, GMOVE, GMERGE and GFILL. A maximum of 12 points maybe entered in each GLINE, but multiple GLINE calls may be made and the resulting strings "added" together to form a single line description containing over 12 points.

Parameters:

1. The name of the string into which the line description is to be placed.

2 and 3, 4 and 5 etc. Descriptions of each of the points in the line where the first parameter is the pixel row and the second is the pixel column in each of the parameter pairs.

Further descriptions of each of the points can be made as follows:

Add 512 to column:

This defines the point but do not display it. This can be used when a standard start reference point to a picture is required e.g. the top left hand corner of a special character.

Add 1024 to column:

Stops a line being drawn from the previous point. A line is drawn from the previous point to this point otherwise.

Add 8192 to column:

Defines that the line drawn from the previous point was a RIGHT boundary.

Add 16384 to column:

Defines that the line drawn from the previous point was a left boundary.

For a line description to be accepted as a boundary at least one line must be defined as, either a left or right boundary. Horizontal lines MUST be described as NEITHER a left or right boundary. Points on a line may be both a left and a right boundary.

A line joining the points (5,20), (2,60) , (12,60) and (12,17) could be described as a line by

```
M GLINE,A$,5,20,2,60,12,60,12,17,5,20
```

For this to be a complete boundary description the macro call could be

```
M GLINE,A$,5,20,2,60+16384,12,60+8192,
-12,17+8192,5,20+8192
```

or

```
M GLINE,A$,5,20,2,16444,12,8252,12,
-8209,5,9212
```

The points may be described using integer variables.

4.15. GMERGE

Function:

This macro merges the area within a boundary on the other graphic screen onto the current graphics screen. The pixels on the current screen have priority and if any pixels within a group of six are on in a different colour, the pixels on the other screen are not merged. If a group of 6 pixels have the same colour, then any

pixels on the other screen are added to those on the current screen. If all pixels in a group of 6 pixels on the current screent are all off, then whatever is on the other screen is added.

If the string contains a line description, it is converted into the Last Defined Boundary (LDB) form. (In this case GMERGE uses the other graphics screen as a work area and will draw a boundary in the current colour on that screen.This can be avoided by converting it first using GCONVERT.)

Parameters:

1. The name of the string containing the boundary description. If this is not in Last Defined Boundary form, an error occurs. If LDB is specified, then the last boundary used is taken.

2. Pixel row in which the first point is to be placed. This overrides any stored in the string and places the boundary in the new position.

3. Pixel column.

4.16. GMOVE

Function:

This moves the area within the boundary on the current screen into the new area on the current screen. The boundary points are part of the area. If the boundary is defined a line description, it is converted into the LDB form first. This conversion uses the other graphics screen as a work area. (see GCONVERT.) If the string does not contain a valid boundary, an error occurs. In the move, the current area is cleared to black i.e. all pixels are switched OFF.

Parameters:

1. String variable containing the boundary description. If LDB is specified the last used boundary definition is taken.
2. Current row position of boundary (Default as defined in the string).
3. Current column position of boundary (Default as defined in the string).
4. New row position of boundary (No default).
5. New column position of boundary (No default).

The rows and columns for either definition may be outside the screen, i.e. they may be negative or greater than the maximum in a row or column.

4.17. GPOINT

Function:

Returns the colour of the specified point on the current graphics screen.

Parameters:

1. Integer variable into which the colour is to be returned where

    0 = Blue
    1 = Green
    2 = Red
    3 = White
    4 = OFF

2. Pixel row of the point.

3. Pixel column of the point.

4.18. GPRINT

Function:

Prints onto the current graphics screen, in the current graphics colour, a picture or line description. This may either be held in a string or on a disk file. The description may be held in either one of 3 forms -

(a) As a line description set up by GLINE. This is as a series of points with lines joining them. (See GLINE for description).

(b) In an LDB form. This is as produced by GCONVERT, GSWAP, GMERGE, GMOVE or GFILL. This takes more memory to store. For these, GPRINT draws a complete boundary line irrespective of the line drawing instructions given in GLINE.

(c) As a Screen Dump. This is created by a utility from a screen created by a lightpen or digitiser and will usually be picked up from disk, or may be created by GSAVE and saved in a string. The string or disk record contain a 4 byte header followed by the screen dumps of rows on the screen where each byte describes 6 pixels. The length of the row is held in byte 4 and is determined when the dump is created.

GPRINT may be used to read a file, created by GSAVE, into a string and print it if required. A screen dump may be printed without storing it in a string.

Parameters:

1. Name of string containing the line or picture definition. If not specified, parameter 4 must be given. If LDB is specified and not parameter 4, then the last used boundary definition is taken. If LINE is specified, then the remaining parameters for GPRINT are as set out for parameters 2 onwards in GLINE, and the

line is drawn but not stored.

2. Pixel row of where initial point is to be placed.

3. Pixel column of where initial point is to be placed. Both 2 and 3 may be negative or outside the screen.

4. Filename containing string. If present, parameter 1 may contain the name of a string variable into which the string is to be stored if the user wishes to reuse it without reloading it from disk each time. A screen dump may take up to 16K for a 480 screen, and 8K for a 240 screen. Storage of these in strings should be used with discretion. The file must have been saved using GSAVE.

5. NO if the string read from the file is not to be printed.

4.19. GRESET

Function:

Switches a pixel off on the current graphics screen.

Parameters:

1.Pixel row.
2.Pixel column.

Function:

To save a portion of a graphics screen in a string, as a SCREEN DUMP. The area saved must be rectangular. GPRINT may then be used to print the diagram in a different place on the screen in the same colour. Remember though, that it takes 8K bytes to save a complete 240 screen.

To save a string onto disk for later retrieval by GPRINT. To dump to disk, the GSAVE macro call, creates a string the same length as the amount of screen being dumped in user memory, therefore requiring 8K of string space to dump an entire screen.

Parameters:

1. Name of the string into which the dump is to be saved. If the filename is specified, this parameter is not necessary.
2. Start pixel row.
3. Start Teletext character column.
4. End pixel row.
5. End Teletext character column.
6. Name of the file into which the string is to saved.

If the filename is specified and no string name, then the screen dump is saved directly onto disk.
If only the string name and the filename are given, then the contents of the string are saved onto disk.

4.21. GSET

Function:

To switch on a particular pixel on the current screen in the current graphics colour. If the 6 pixel group is of the same colour as the current colour, then the particular pixel is switched on. If the 6 pixel group is of another colour, then the whole 6 pixel group is switched to OFF, the colour changed and only the defined pixel is switched ON. This also applies when drawing lines.

Parameters:

1. Pixel row

2. Pixel column

4.22. GSECTOR

Function:

Creates a line description of a circle sector, suitable for use with GFILL. All parameters are required. Parameters 5 & 6 must have values between 0 and 1. Where position 0 is at the bottom of the circle, and 1 is a complete circle. For example, to draw the right half of a circle, parameters 5 & 6 would be 0 and .5. As this routine is written in Basic, it is not very fast. The circle is defined in an anti-clockwise direction.

Parameters:

1. Name of string for desciption to be placed in.

2. Pixel row for centre.

3. Pixel column for centre.

4. Radius in horizontal pixels.

5. Start point of sector.

6. Size of sector. GSWAP

Function:

Swaps the areas within the boundary on the two graphics screens. If the boundary is not entered then the whole screen areas are swapped. If LDB is entered, then the Last Defined Boundary is used.

Parameters:

1. Name of string containing the boundary.
2. Start pixel row.
3. Start pixel column. If either of parameters 2 or 3 are not entered, then the start position as defined in the boundary is used.

4.23. GWIPE

Function:

To draw a line in black - that is to switch off all the pixels which make up a line.

Parameters:

All the parameters are exactly as defined for GPRINT.

4.24. INDIGIT

Function:

INDIGIT allows for the input of numeric values with an option for the decimal point. No non-numeric characters are accepted including (© E + * / -). A string is returned, as well as Z which has the VAL of the string. This input can be terminated by ENTER as in a regular Basic input.

Parameters:

1. String variable name for the input to be placed in.

2. Length of the string to be inputted.

3. Start row of string (cursor position).

4. Start column of string (cursor position).

5. Decimal point option. Set to 1 for decimal point. Defaults to no point.

4.25. INTEXT

Function:

INTEXT allows for the input of strings without control characters, the length of the string being specified in the Macro call. This input can be terminated by a ENTER as for a regular Basic input.

Parameters:

1. String variable name for the input to be placed in.

2. Length of the string to be inputted.

3. Start row of string (cursor position).

4. Start column of string (cursor position).

4.26. LINE

Function:

LINE draws a line between any 2 points on the current teletext screen. The relevant chunky graphics are turned on in whatever is the current colour defined for that byte. If the byte contains a non-graphics character, then it is replaced by a byte containing the one chunky graphic turned on.

NOTE: This routine cannot determine whether or not a particular character is in graphics mode. It is up to the programmer to ensure that it is.

Parameters:

1. Chunky pixel row (From)

2. Chunky pixel column (From)

3. Chunky pixel row (To)

4. Chunky pixel column (To)

4.27. LOGOFF

Function:

To log a user off the computer and return to the log on screen.

Parameters:

none.

4.28. LPRINT

Function:

To print to the line printer. The first time this macro is used, the lineprinter is OPENED. It is CLOSED automatically at the end of the program.

Parameters:

1. String or strings to be printed

2. String or strings to be printed

3. String or strings to be printed

4.29. MIX

Function:

The screens are normally displayed in the priority:

T1, G1, T2, G2, Background,

where any pixel displayed on a higher priority screen, has display priority. In MIX mode, T1 still has the highest priority and Background the lowest priority, but the Screens G1, T2 and G2 are mixed. I.e. If the Red beam is on in any one of the 3 screens for a particular pixel, it is displayed. Similarly for the Green and Blue beams. Hence the colours Magenta, Yellow and Cyan can be obtained by mixing.

Parameters:

1. MIX or PRIORITY - Default PRIORITY.

4.30. MOVE

Function:

To copy a block of data in memory.

Parameters:

1. Start address of block to be copied.

2. End address of block to be copied.

3. New start address.

4.31. NEXT

Function:

Sets up a screen exit routine for either moving onto the next screen, back to a previous point in the program, or to any other point. It displays a message on line 23 and accepts a one character input. A right arrow means continue onto the next screen, a left arrow means go back to the previous screen, and other characters are as the user defines.
While waiting for the character, a subroutine may be performed.

Parameters:

1. Name of the procedure to go back to on pressing the left arrow cursor key. If not given, the backstep option is not included.
2. Decimal value of another key pressed OR the value SUB. If SUB is specified then parameter 3 must contain the name of the subroutine to be performed while waiting for a character.
3. Procedure to go to when that key is pressed OR if parameter 1 is SUB, the name of the subroutine to be performed. The subroutine must either RETURN to

selector regularly so that it can check for a key press OR the subroutine itself may check for a key press and only RETURN when one is pressed. The RETURN to selector should be made if (USR(0)AND256)<>0.

4 to 19: Further pairs of parameters similiar to parameters 2 and 3.

4.32. PAUSE

Function:

To enable and disable the PAUSE key. This key must be disabled during any operation requiring timing.

Parameters:

1. ON or OFF

4.33. POINT

Function:

To determine whether a particular teletext graphic is ON or OFF. If the point is ON, 1 is returned and if OFF 0 is returned.

Parameters:

1. Integer variable into which the result is to be placed.

2. Chunky pixel row

3. Chunky pixel column.

4.34. PRINT23

Function:

To print on line 23 of the current teletext screen.

Parameters:

1. String to be printed.

4.35. RANDOM

Function:

This resets the random number generator to start a new random number series.

Parameters:

None.

4.36. READCLOCK

Function:

To get the current time as an 8 character string ie. HH:MM:SS

Parameters:

1. Name of string into which time is to be placed.

4.37. RESET

Function:

To turn a particular teletext chunky graphics character OFF on the current teletext screen.

Parameters:

1. Chunky graphics pixel row.
   OR the value INIT. If INIT is specified, then the first character in every row on the current teletext screen is replaced by a space.

2. Chunky graphics pixel row.

4.38. REVERSE

Function:

To set the reverse video on the current teletext screen. Any data written to the current teletext screen while REVERSE is set, is written in REVERSE video, i.e. the foreground colours and background colours are each written in their respective opposite colours.

Parameters:

1. ON or OFF

4.39. RUNCLOCK

Function:

The function restarts the clock from the point at which it was stopped.

Parameters:

None.

4.40. SCHAR

Function:

This is a special print routine which displays a special character on the current graphics screen in the current graphics colour. The position it is to be placed in, is given as the teletext character row and column. The special character is defined using a line description with the top left hand pixel defined as the reference point, with that point not being printed. (see GLINE). Each teletext character occupies a 6 by 10 block of pixels. The left hand column is always blank. The last 3 rows become the space between the lines.

Parameters:

1. String containing the special character as a line description.

2. The teletext character row.

3. The teletext character columns.

4.41. SCREEN

Function:

SCREEN is used to set the current screen type for both teletext and graphics. One teletext and one graphics screen are current at any one time. At the start of a program, T1 and G1 are the current screens. The current screen is the screen on which any printing is done when PRINT and GPRINT are used.

SCREEN also sets the mode in which a screen is to be displayed. T1 can be set to 20, 40 and 80 mode. It is initially set to 40. G1 may be set to either 240 or 480 mode. It is initially set to 240. T2 and G2 are fixed at 40 and 240 respectively. When T1 is set to 80, T2 is inoperative. When G1 is set to 480, G2 is inoperative.

Parameters:

1. Screen to be selected as the current screen. T1, T2, G1 or G2.

2. Mode in which screen is to be set. This may be 20, 40 or 80 for T1 and 240 or 480 for G1. If the mode is not specified, the previous mode set remains unchanged.

4.42. SELECTOR

Function:

SELECTOR may be used to set up to 12 selection boxes on the current Teletext screen and activates a cursor which can be moved from box to box. The cursor is controlled by the four cursor arrows. When the user has the cursor in the box which he wishes to select, he presses <ENTER> to make the selection. For the purposes of specifying the position of the boxes, the screen is

divided into an 7 by 8 grid, each box covering 5 Teletext columns and 3 Teletext rows. These are numbered from 0 through to 55. There are 7 rows of 8 boxes. No boxes are placed in the last three teletext rows. Each box may be preceded by as many lines of text as required, or if only a single character selection is to be made, the character may be displayed inside the box. The single character must have a ASCII value between decimal 64 and 95. Each box covers 3 Teletext rows and 3 Teletext columns. Boxes may be placed in adjacent Selector rows and columns if required. So that the cursor controls can be used, the programmer must ensure that the boxes are positioned with either horizontal or vertical movement, or both are possible. The number of the selected box is returned in Z%, the boxes being numbered from 1 in the order in which they were defined. The Teletext screen is NOT cleared either before or after. A maximum of twelve selector boxes may be defined on one screen. When selction has been made, the selected box is left with the centre ON. A Selector screen may be reentered without redrawing (see Parameter 1). Exit from the screen may also be made on user specified values (see macro SELECTVAL).

Parameters:

1. The colour the text is to be printed. (RED, GREEN, BLUE, YELLOW, CYAN, MAGENTA, WHITE) Default WHITE. Further colour specifications may be placed within the string using Teletext colour control characters. Care must be taken to ensure that the first character in this case is not taken as a line length.

OR

The value REENTER if a previously drawn screen is to be reentered. Previous selections remain illuminated. the programmer must ensure that the screen has not been altered in the interim.

2. The colour the box is to be printed. (Choice and default as for parameter 1.)

3 to 26. The remaining parameters are grouped in pairs

where:

(i) Selector box position number. Remember that the text is placed BEFORE the box. The box is printed in the last three Teletext columns of the 5 columns in the box.

(ii) Either the name of a string variable, or a literal string, which conta ins the text to be printed before the box.

If the string contains a single character in the range @ to # (i.e. decimal 64 to 95), then that character is placed within the box and flashed instead of the cursor. In this case no text is placed prior to the box so that boxes may be placed in adjacent horizontal positions.

If the text is to be placed on a single line, the string contains just that text and the text is placed on the top line of the box, before the box with a single space separating the box and the last character of the text.

If multiple lines of text are required, then the complete text is put into the string preceded by a single character indicating the length of each line of the text. The maximum line length is 31 characters. e.g. If 3 lines each of 5 characters are required, then the string would be CHR$(5)+"Line1Line2Line3".

4.43. SELECTVAL

Function:

SELECTVAL sets up alternative values on which returns are made from the Selector screens. The values entered in this macro affect ALL Selector screens set up in the program.

Parameters:

Up to 8 parameters may be entered each one containing an ASCII decimal value on which exit is to be made from the Selector screens. The variable Z% contains the value on which the exit was made rather than the box number in this case. As values entered should all be greater than 12 there should be no confusion.

4.44. SET

Function:

SET sets a particular teletext chunky graphics pixel on. Each teletext character is divided into 6 pixels. To use graphics mode, the graphics control character must be set on prior to the use of graphics in each row. Column 0 is therefore not availble for graphics. Colour must also be redefined on each row. If a pixel is required to be other than white, the colour control character must be inserted. Colour graphics other than white are not therefore able to be placed in column 1.

This function does not set up colours. Any pixel set on will be displayed in the colour defined at that point. Similarly, this function does not determine whether graphics mode is set or not. If the code in a particular character is not a graphics code, ie between 32 and 63 or between 96 and 127, it is replaced by a character containing just the pixel required set ON.

This function may also be used to put a graphics control code in column 0 of each row.

Parameters:

1. Chunky pixel row or INIT. If INIT is specified, then a graphics control code (30) is inserted in column 0 of every row.

2. Chunky pixel column.

4.45. SETCLOCK

Function:

To set the clock to a given time.

Parameters:

1. String containing the time in the form

hh:mm:ss

where hh = hours
mm = minutes
ss = seconds.

The user may use any delimiter.

4.46. SHOWCLOCK

Function:

To display the clock on the screen each second. The display is placed in the last 8 columns of row 23 unless changed using the parameters in this function. The display may also be turned off using this function.

Parameters:

1. Teletext character row for display if current position is not suitable.
or OFF to suppress display.

2. Teletext character column.

4.47. SKEL

Function:

To provide a standard BASIC program skeleton. The main code is placed in

P MAIN. Other positions defined include

INIT for initialization routines for the heading screen

INIT1 for the remaining initializtion routines

OPEN for opening of disk and printer files

CLOSE for closing of disk and printer files

FINISH for end of program processing

SUB for subroutines

Parameters:

1. Program Name.

2. Program title line 1 - inverse, double height.

3. program title line 2 - inverse, double height.

4. Written by - documentation only.

5. Programmed by - documentation only.

6. Next Program in chain. A space will cause no CHAIN to be generated. (Default - return to the menu program)

7. Subtitle #1 - displayed on the heading screen on row 11 in green characters.

8. Subtitle #2 - as subtitle #1 but displayed on row 13.

9. Subtitle #3 - as subtitle #1 but displayed on line 15.

10. No of additional screens required for graphics (Default 0)

11. Chaining program message. Displayed when chaining next program.

4.48. STOPCLOCK

Function:

To stop the clock running. This also switches off the clock display.

Parameters:

None.

4.49. TEXTGET

Function:

Gets a string of text from the current teletext screen.

Parameters:

1. Name of the string into which the text is to be placed.
2. Start row.
3. Start column.
4 Number of characters to be taken from the screen.

4.50. TIME

Function:

TIME returns the time in 10 millisecond intervals (100ths of seconds). The value returned is an integer between -32768 and 32767. This is continually updated and on reaching 32767, returns to -32768. This occurs aboutevery 10 minutes.

Parameters:

1. Integer variable into which the time is to be placed.

4.51. WAIT

Function:

To halt processing for the specified number of 10 millisecond (100ths of seconds) intervals.

Parameters:

1. The number of 10 millisecond intervals to WAIT.

5. USEFUL USER CALLS AND POKES

5.1. Initialising USR functions < 256

The USR functions < 256 may only be used in BASIC if

```
POKE 1182,1
```

is done first. This is done in the skeleton program.

5.2. Teletext 80/20 Colour

```
Z% = USR(18, USR(19) AND NOT 1536 OR nnnn)
```

where

nnnn = 0 for BLUE
nnnn = 512 for GREEN
nnnn = 1024 for RED
nnnn = 1536 for WHITE

5.3. Background Colour

```
Z% = USR(18, USR(19) AND NOT 112 OR nnn)
```

where

nnn = 0 for NONE
nnn = 16 for RED
nnn = 32 for GREEN
nnn = 48 for YELLOW
nnn = 64 for BLUE
nnn = 80 for MAGENTA
nnn = 96 for CYAN
nnn = 112 for WHITE

These are all displayed at half intensity.

5.4. Graphics Colour

POKE 1161,n

where

n = 0 for BLUE
n = 1 for GREEN
n = 2 for RED
n = 3 for WHITE

5.5. Displaying Screens

Z% = USR(18, USR(19) OR nnnn)

to switch a screen ON.

Z% = USR(18, USR(19) AND NOT nnnn)

to switch a screen OFF

where

nnnn = 8 for Teletext 1
nnnn = 1 for Teletext 2
nnnn = 8192 for Graphics 1
nnnn = 2 for Graphics 2.

5.6. Selecting Current Screens

```
Z% = USR(46, n)
```

where

n = 1 for Teletext 1
n = 2 for Teletext 2

```
POKE 1160, n
```

where

n = 0 for Graphics 1 (240)
n = 1 for Graphics 1 (480)
n = 2 for Graphics 2.

5.7. Selecting Screen Mode

```
Z% = USR(18, USR(19) AND NOT nnnn OR xxxx)
```

where for

Teletext 80

```
nnnn = 6145 and xxxx = 6144
```

Teletext 40

```
nnnn = 6144 and xxxx = 2048
```

Teletext 20

```
nnnn = 6144 and xxxx = 4096
```

Graphics 240

```
nnnn = 4 and xxxx = 0
```

Graphics 480

```
nnnn = 0 and xxxx = 4
```

NOTE: For 480 mode an additional Graphics area must have been selected from the SKEL macro or by

```
USR(1211, n)
```

where n indicates the number of additional screen areas. This MUST be done first thing in a program.

5.8. Selecting 12 and 24 line screens

```
Z% = USR(nn)
```

where

nn = 47 for 24 line screens
nn = 48 for 12 line screens

Further the top and bottom pages of the 12 line screens may be selected with

nn = 49 for the top page
nn = 50 for the bottom page.

5.9. Selecting MIX/PRIORITY

```
Z% = USR(18, USR(19) XXXXX 16384)
```

where

XXXXX = AND NOT for Priority
XXXXX = OR for Mix.

5.10. Reverse Video

```
Z% = USR(11, n)
```

where

n = 0 to disable it
n = 1 to enable it.

Individual characters may be set to reverse video by adding 128 to their decimal value.

5.11. The current graphics screen is cleared by

```
Z = USR(1217,0)
```

5.12. Graphics Subroutines

The entry points for the graphics subroutines are between 1211 and 1233. The parameters required vary. The addresses are:

GRAPHICS (1211) Sets the screen areas. Must be at the start of a program.
GSET (1213) Sets a pixel ON or OFF.
GPOINT (1215) Returns the status of a pixel.
GPRINT (1217) Prints a string.
GMOVE (1219) Moves the area within a boundary.
GFILL (1221) Fills the area within a boundary.
GMERGE (1223) Merges the 2 screens.
GSWAP (1225) Swaps the screens.
GCONVERT1 (1227) Saves the LDB area in a string.
GSAVE (1229) Saves a screen area in a string.
GCONVERT2 (1231) Converts a line description into a LDB.
GCIRCLE (1233) Draws a circle.

5.13. Disabling the PAUSE KEY

This is accomplished by

```
POKE 1183,0
```

and enabled by

```
POKE 1183,1
```

5.14. Disabling the BREAK KEY

This is accomplished by

```
POKE 1184,0
```

and enabled by

```
POKE 1184,1
```

5.15. Program Communication

The CHAIN function does not preserve the program variables so that information to be handed between programs must be either written to disk or written to a protected area of memory. 17 bytes are reserved for use by programmers. These are at locations

```
1165 to 1181
```