ARcalc

An Amiga Calculator
Version 1.1

by Roberto Attias

1 Introduction

ARcalc is a program, developed for the Amiga series of computers, that emulates a scientific calculator. ARcalc can be used with any Amiga model running a version of the operating system greater then 1.3.

Version 2.1 of the operating system introduces a localization feature which allows programmers to write programs that are able to comunicate with the user in the language he chooses. ARcalc is able to make use of this feature, and is distributed with the files needed to localize the calculator menu items and error messages in a number of languages. I decided not to localize the button labels, because even on a real calculator these are not usually localized.

ARcalc provides almost every feature you might expect from a scientific calculator, including binary, hexadecimal and octal base calculation. The trigonometric functions can be applied to values measured in degrees, radians, or decimal degrees; numbers can be rappresented in scientific notation, with fixed decimal point (allowing you to choose the number of decimal digits), or as a combination of formats. Every computation is performed with an internal double precision.

The current release of ARcalc is version 1.1. Version 1.0 was never released, and was not localized.

Remember that ARcalc **does not** work with versions of the operating system lower than 2.0; if you try to start it using an earlier version of the operating system, nothing will happen.

2 Using ARcalc

The program can be started from Workbench or Shell. It looks like a standard calculator, sitting on the Workbench screen, in which you should recognise the display and the keyboard. To avoid confusion, the word 'keys' will be used to indicate the Amiga keyboard keys, while the word 'buttons' will be used to indicate the ARcalc keyboard keys.

You can vary the font and font size used to display ARcalc. The font used, as many other initial characteristic that we will see later, is defined in a configuration file called ARcalc.config that would have to resides in the same directory as the ARcalc program. If this file is not present ARcalc uses the same font as Workbench. Starting from Shell, you can specify an alternate font by typing:

ARcalc <font name> <Y dimension>

ARcalc will override the config file default, and will try to open the specified font. The font chosen is used for the keyboard labels and for the display, but not for the ARcalc menus. It is also possible to change the font after the calculator has been started, using the Set font ... item in the Settings submenu of the Project menu. (See Chapter 4 [ARcalc menus], page 6.)

ARcalc boot can fail for a number of reasons (not enough memory, font chosen too big, errors opening libraries..); if this happens the user is informed with an appropriate error message.

The main feature of ARcalc is the ability to type complex expressions in the same way they are usually written. This can be done using either the Amiga keyboard or the ARcalc

keyboard. You can see the expression you are typing at the top of the display; this is called the "edit area". You can edit the expression using either the tipical Amiga string gadget shortcuts or the special `ARcalc` editing buttons.

To calculate the result of the given expression, you have only to press the `RETURN` key, or to click the `=` button; the result will be displayed on the right side of the bottom of the display. This is called the "result area". `ARcalc` uses the result area also to show you its messages about syntax or miscalculation errors.

Note that if a syntax error is found in the typed expression, the cursor of the edit area indicates where the error is, while if a miscalculation occurs (e.g. a division by zero) the cursor position is meaningless. For a complete list of the syntax/miscalculation errors, see Chapter 4 [ARcalc menus], page 6, of this manual.

3 Detailed description of ARcalc buttons

Let's now see the meaning of `ARcalc` buttons. There are four button types:

3.1 Editing buttons

You can find this kind of button in the upper right part of the calculator. They are:

`<=` (left arrow): allows you to move the cursor of the edit area one character left. You can do the same thing using the corresponding Amiga keybord key.

`=>` (right arrow): allows you to move the cursor of the edit area one character right. You can do the same thing using the corresponding Amiga keybord key.

`Del` (Delete): allows you to delete the character the cursor is over in the edit area. You can do the same thing using the Amiga `Del` key.

`Bck` (Backspace): allows you to delete the character at the left of the cursor in the edit area. You can do the same thing using the Amiga `<-` (backspace) key.

`Clr` (Clear): allows you to clear the edit area, loosing the expression you was editing. You can do the same thing with the `Amiga-Left X` shortcut.

3.2 Status buttons

This kind of button modifies the status of the calculator. The status is showed as a set of abbreviations in the lower left part of the display; this is called the "status area". Note that the global state when you start `ARcalc` depends on the configuration file `ARcalc.config`; if this file is not found, default values are assumed.

The status buttons are the following:

`Bse` (numeric BaSE): allows you to choose the base in which you are going to work, between binary, octal, decimal or hexadecimal. The actual setting is shown in the status area, by one of the following abbreviations:

- `dec` decimal base
- `hex` hexadecimal base

`oct` octal base

`bin` binary base

Pressing the `Bse` repeatedly cycles through the bases, in the following order:

`... -> dec -> hex -> bin -> oct -> ...`

When you change base, the actual expression is automatically cleared. This is because every number in the expression would change it's value in the new base. When the base is not hexadecimal, certain buttons become redundant (e.g. the `A`, `B`, `C`, `D`, `E`, `F` keys, if you are working with decimal base).

Also note that `ARcalc` is able to use real numbers only in decimal base; if you work in a different base you must use only integer numbers. This implies that many functions, although still available, are meaningless (e.g. the `sin(x)` function still works, but x must be integer, and it returns only 0 or 1). If the `ARcalc.config` file is not present when starting, the decimal base is used by default.

`Fse` (Fixed,Scientific,normal): this button allows you to choose the notation in which the results are shown. Pressing `Fse` repeatedly cycles through the following options:

`Fixed`: indicated by the `fix` abbreviation; the results are shown as numbers with a fixed number of decimal digits. You can fix the number of decimal digits using the `Fix` button.

`Scientific`: indicated by the `sci` abbreviation; the results are shown in the scientific form. Note that the number of digits fixed with the `Fix` button is not longer related to the number of decimal digits, instead it indicates the number of significant digits.

`normal`: indicated by the `nor` abbreviation; the results are shown using whichever of the above notations generates the shortest string.

Note that by changing format, the result shown in the result area changes its format also. If the `ARcalc.config` file is not present when starting, the normal format is used by default.

`Drg` (Degrees, Radiant, Decimal Degrees (Grad)): this button allows you to choose the measuring system unit for arcs in trigonometrical functions. Pressing `Drg` repeatedly cycles through the following options:

- `Degrees`: indicated by the `deg` abbreviation.
- `Radians`: indicated by the `rad` abbreviation.
- `Decimal Degrees`: indicated by the `grad` abbreviation.

If the `ARcalc.config` file is not present when starting, radians are used by default.

`Fix` (fix significant digits): this button allows you to fix the number of significant digits or decimal digits, depending on the chosen format. When you press this button in the result area you will see the following display:

`Set number of digits (0 - 9)`

Now you should press one of the digit buttons of `ARcalc`, from `0` to `9`, fixing in this way the number of digits; if you press a different button, nothing happen, the operation is aborted keeping the same number of digits as before. Note that it does not matter how many digits you fix: every calculation is performed in an internal double precision format. If the `ARcalc.config` file is not present when starting, the number of digits is fixed to 9 by default.

`Inv` (Inverse function): this button allows you to obtain the inverse functions. When you press it, the `inv` abbreviation is displayed in the status area: if the next button you press indicates a function, and if is possible to invert this function, then you will get it; otherwise the `Inv` press produces no effects.

`Hyp` (Hyperbolic function): this button allows you to obtain the hyperbolic functions. When you press it, in the status area you can see the `Hyp` abbreviation: if the next button you press indicates a trigonometrical function, then you will get its corresponding hyperbolic function; otherwise the `Hyp` pression produces no effects.

3.3 Text buttons

This kind of button allows you to insert digits, functions, etc. in the expression. You can also do this using the Amiga keyboard, but these buttons offer a convenient shortcut. Note that the expression parser is not case sensitive, so you can type it in upper or lower case. The text buttons are:

`log` if pressed alone produces the string `Log(`, indicating the *logarithm* function (using base 10); if you press `Inv` before, you obtain the string `10^`.

`ln` if pressed alone produces the string `Ln(`, indicating the *natural logarithm* function (using base e); if you press `Inv` before, you obtain the string `e^`.

`x2` if pressed alone produces the string `^2`, indicating the square; if you press `Inv` before, you obtain the string `Sqrt(` indicating the *squareroot* function.

`mod` when pressed, produces the string `Mod(` indicating the function *modulus*. '`Mod(a,b)`' is the remainder of the division a/b.

`sin` if pressed alone produces the string `Sin(` indicating the trigonometric function *sine*; if you press `Inv` before, you obtain the string `ArcSin(`, indicating the *arcsine* function; note that result of these functions depends on the measuring system unit chosen. If you press `Hyp` before, you obtain the string `Sh(` or `ArcSh(`, indicating the *hyperbolic sine* and its inverse function;

`cos` if pressed alone produces the string `Cos(`, indicating the trigonometric function *cosine*; if you press `Inv` before, you obtain the string `ArcCos(` indicating the *arccosine* function; note that result of these functions depends on the measuring system unit chosen. If you press `Hyp` before, you obtain the string `Ch(` or `ArcCh(` indicating the *hyperbolic cosine* and its inverse function;

`tan` if pressed alone produces the string `Tan(`, indicating the trigonometric function *tangent*; if you press `Inv` before, you obtain the string `ArcTan(` indicating the

arctangent function; note that result of these functions depends on the measuring system unit chosen. If you press `Hyp` before, you obtain the string `Th(` or `ArcTh(` indicating the *hyperbolic tangent* and its inverse function;

`and` when pressed, produces the string `and` indicating the binary *and* operator. The operand are implicitely forced to 32-bit integers, before performing the operation.

`or` when pressed, produces the string `or` indicating the binary *or* operator. The operand are implicitely forced to 32-bit integers, before performing the operation.

`not` when pressed, produces the string `not` indicating the binary *not* pre-operator. The operand are implicitely forced to 32-bit integers, before performing the operation.

`xor` when pressed, produces the string `xor` indicating the binary *xor* operator. The operand are implicitely forced to 32-bit integers, before performing the operation.

`fct` when pressed, produces the string `Fact(` indicating the *factorial* function.

`rnd` when pressed, produces the string `Rnd(` indicating a *one-parameter* function, for the generation of a pseudo-random number. If x is the value of the parameter, the number will be generated in the range `[0,x]` if x>0, in the range `[x,0]` otherwise.

`( & )` these buttons produce the corresponding character.

`pi` this button produces the string `Pi` indicating the symbolic constant `PI`, whose internal value is `3.14159265358979323846`

`e` this button produces the string `e` indicating the symbolic constant `e` (Napier's constant), whose internal value is `2.718281828459045`.

`0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F`
when pressed,these buttons produce the corresponding character, indicating a digit. Obviously, when you compose an expression, the number interpretation depend on the numeric base you have set. Note that all these keys will always work, even if in some bases they should not be used.

`Exp` this button produces the `E` character, that you have to use to type a number in exponential form. This character is not different from the e indicating the Napier's constant, or from the `E` digit in hexadecimal base; its interpretation depend on the context in which it is found.

`,` this button produces the corresponding character, needed to separate the arguments of a function. Only the `Mod(a,b)` function has more than one argument, so you may feel that using keyboard is sufficient, however this button has been included to preserve the ability to use the calculator only by mouse.

`+,-,*,/,^`
these buttons produce the corresponding character, indicating the *sum*, *difference, product, ratio* and *power* operators. Note that the minus sign can also be used as a preoperator to modify the sign of an operator.

`v,w,x,y,z`
: these buttons produce the corresponding character, indicating one of the five `ARcalc` variables. If, during the expression evaluation, `ARcalc` finds one of this variables, it use its current value. To learn how to assign a value to a variable, see the description of the `sto` button.

3.4 Special buttons

This class consist of the buttons with special functionalities. These buttons are:

`sto`
: allows you to store the value of the expression currently beingin edited in a variable. When you press this button, the following message is shown in the value area:

  `choose var. to store:`

  You should now press one of the five variable buttons: if you do, the expression will be evaluated, and the result will be assigned to the variable you chose. If you press any other button, instead of a variable one, there is no effect.

`cpy`
: this button allows you to insert, in the expression you are editing, the value of the preceding evaluation, visible in the value area.

`get`
: this buttons allows you to copy a text from the Amiga clipboard, to the edit area. In this way you can cut an expression from an application, and inport it to `ARcalc`, ready to be evaluated. You can do the same using the `Paste Expression` item in the `Clipboard` menu.

`put`
: this buttons allows you to copy the value visible in the value area to the Amiga clipboard (as a string). In this way you can cut the result of an evaluation from `ARcalc`, and paste it in an other application. You can do the same using the `Cut Result` item in the `Clipboard` menu.

`=`
: by pressing this button, you start the expression evaluation. If no syntax/evaluation errors are encountered, the result will be shown in the value area, otherwise an error message will be shown in the same area. Note that, depending on the `AutoCLR` flag, the expression text may or may not be cleared after the evaluation. The `AutoCLR` flag value can be changed using the `AutoCLR` item in the `Setting` submenu of the `Project` menu (see Chapter 4 [ARcalc menus], page 6.)

4 ARcalc menus

`ARcalc` has two menus, named as `Project` and `Clipboard` (using english localization).

The '`Project`' menu has the following items:

`Settings`
: this is a submenu with items as follows:

  `Set Font ...`
  : allows you to modify the font actually used to show the calculator, using a standard Amiga font requester. If the font you choose is too big, an error requester appears, informing you about it. Note

that the new font will not be used the next time you start `ARcalc` if you do not use the `Save defaults` option.

`Use WB font`
: this allows you to use the Workbench font, instead of one chosen with the preceding item. Note that this is different from using the previous item and specifying the Workbench font. With this item, if you save the defaults, when the calculator is started it looks for the current Workbench font, that might have been changed from the last time you used `ARcalc`.

`AutoCLR` this item acts like a switch, allowing to set the `AutoCLR` flag on or off. If the flag is on, after every evaluation, preformed with the `=` or `sto` buttons, or with the `RETURN` key, the text of the expression in the edit area is cleared. If this flag is off, the text is not cleared, allowing to make changes to it and evaluate again.

`Save defaults`
: allows you to save some calculator settings in a file named `ARcalc.config`. When `ARcalc` is started, it looks for this file, and adopts the settings specified within it. The following settings are saved:

- Position of the calculator on the screen
- Font (and Y dimension) used.
- `AutoCLR` flag status
- `ARcalc` status (numeric base, arc measuring system unit, numeric format, number of significant digits).

`About ...` it shows a requester with informations about the program and its author. The same function is performed by the `Inf` button in `ARcalc`.

`Quit` quits the program. The same effect is obtained pressing the `ARcalc` CloseWindow gadget.

The `Clipboard` menu has the following items:

`Cut Result`
: copies the most recently evaluated result to the Amiga clipboard. The same effect is obtained pressing the `ARcalc put` button.

`Paste Expression`
: copies text from the Amiga clipboard to the edit area, skipping non-alphanumeric characters. The same function is performed by the `ARcalc get` button.

5 Syntax/evaluation errors

Here is a detailed description of `ARcalc` error messages, shown in the result area of the display:

5.1 Syntax errors

- `Invalid Operator`: after a valid operand, `ARcalc` found a sequence of symbols not recognised as a valid operator.
- `Missing digits in number`: there are no digits after a decimal point.
- `Invalid Exponent`: the exponent of a number written in the exponential format, is too big, is missing, or is incorrect.
- `Missing parenthesis`: opened parenthesis has not been closed.
- `Unknown operand`: where an operand is expected, an unknown sequence of symbols is found.
- `Missing Operand`: the expression is ended by an operator.
- `Too many digits`: in the numer there are too many digits.

5.2 Evalutation errors

- `Underflow`: during the evaluation the value has become too small to be represented with a double precision format.
- `Overflow` or `invalid number`: during the evaluation the value has become too big to be rappresented with a double precision format.
- `Division by zero`: the divisor of a division must not be zero
- `Division by zero in Modulus`: the second argument of the Mod function must not to zero
- `ArcSin() argument out of domain`: the argument of this function must be in the range [-1,1]
- `ArcCos() argument out of domain`: the argument of this function must be in the range [-1,1]
- `ArcCh() argument out of domain`: the argument of this function must be >= 1
- `ArcTh() argument out of domain`: the argument of this function must be in the range (-1,1) (excluding the limits themselves)
- `Logarithm argument must be positive`: the Ln() and Log() functions are defined only for positive arguments.
- `Sqrt() argument must not be negative`: Sqrt() function is defined only for argument >= 0
- `Fact() argument must not be negative` the factorial function is defined only for argument >= 0

6 To do in the future

If you like this program, I would like to extend it in the future. I would like to add an `AREXX` port, that will allow other programs to use `ARcalc` for their calculations. I may also write some programs using this feature e.g. a function plotter, or an interpreter for a little calculation language, using `ARcalc` as a calculation host.

Any other suggestions are welcome.

7 Acknowledgements

I wish to thank some people, for the help they have given me:

- Marco Caimi, Federica Colla, Paolo Silvera, Marco Zandonadi, for the beta testing of the program;
- Steve McKinty for the english revision of this manual (I modified something after his revision, so maybe it is not still perfect);
- Reinhard Spisser for converting the doc in TexInfo;
- All my *official localizers*, that made possible to distribute `ARcalc` with so many languages:
  - Steve McKinty (english)
  - Reinhard Spisser (german)
  - Frank Bignone (french)
  - Fer de Jong (dutch)
  - Hannu Helminen (finnish)
  - Anders Hammarquist (swedish)
  - Soeren Berg Hansen (danish)
  - Njaal Eide and Nicolai Langfeldt (norvegian)
- All the "Amiga comunity" of the Computer Science department, University of Milan for their *entusiasm* that helped me much to finish this project.

8 Program and distribution notes

9 How to reach the author

```
e-mail: attias@ghost.sm.dsi.unimi.it

Mail: Roberto Attias
Via Lissoni, 5
I-20162 Milano MI

Phone: +39-(0)2-6470375
```

Table of Contents