Mocca

Mocca is a Frontend for Emc2, the Enhanced Machine Controler.
It is not s Standalone Application, Mocca can only be used as a display- program in the Emc2 environment. Current Version is developed for 3 up to 5 Axis mills. It may work for lathes also but this was not testet on real machines.

Compiling Mocca from source:

As modern applications get more and more complex we decided to develop mocca with a RAD tool called LAZARUS. Lazarus is very similar to Delphi and uses the Freepascal-compiler.

First you have to install the LAZARUS and FREEPASCAL- Packages on your computer. Please refer to

http://www.sourceforge.net/projects/lazarus/
http://www.freepascal.org/

for the installation procedure.

Install Freepascal and Lazarus. You can use the Ubuntu Repository also.

Install the svn (Subversion) package on your computer.

Download and install the emc2 source (Version 2.3x up to current 2.4.1)
Informations how to download and install the Emc2 source you will find on:

www.linuxcnc.org

Compile EMC2 with the –enable-run-in-place option.

Download the lates mocca via svn:

svn checkout http://moccagui.googlecode.com/svn/trunk/ moccagui-read-only

Last, install – if neccesary – the librarys needed by mocca:

libgtkglext1
libglib1.2-dbg
libgdk-pixbuf2

Ok, lets check what we have now:

in /usr/lib we have a folder /lazarus/0.9.28.2/

The Numbers reflect the Version of Lazarus. Please remember this folder as the LAZDIR.

In /usr/share we have a folder /fpcsrc/2.2.4

This is the location of the FreePascal source.

In ~/moccagui/ we have now the following folders:

/src	the source code of mocca
/2.4.1	the Makfiles for Emc2-Versions 2.4.1 and later
/2.3.4	the Makefiles for Emc2-Versions up to 2.3.5

In ~/moccagui/skins we have the following folders:

default	the default Layout and Config for mocca (Refer to DESIGN MOCCA) Remember this location as the CONFIG folder.
hh1280x1024	A sample Layout and Config for 1280x1024 Screens in German.
/pkg	The mocca controls like mocbutton,slider,lister etc...

Next we will try to compile mocca:

change to the ~/moccagui folder that matches your emc2- Version
Lets say you are using emc2-2.4.1

cd ~/moccagui/2.4.1

open the Makefile in this folder with a Text-Editor.

gedit Makefile

Take a look at the following lines of the Makfile:

WM=gtk2

This describes the Widgetset we want to use with mocca. Mocca could be compiled on almost all Platforms. (Windows,MaxOS,Gtk1,Gtk2,Carbon etc...) Off course it does not make sense to compile mocca for Windows cause Windows does not have a Realtime Kernel.
In our case we use the gtk2 Widgetset. Do not change this.

EMCDIR = /home/my_name/emc2-2.4.1

This is the folder where your EMC2-source,binary and libs are located.

For Example:
if you have installed and compiled (with –enable-run-in-place) the source in the folder home/my_name/emc2-2.4.1/ then your EMCDIR is /home/my_name/emc2-2.4.1

please check if the following files are correctly installed:

/lib/librs274.so (.0)
/lib/libemcini.so (.0)
/lib/libemchal.so (.0)

LAZDIR = /usr/lib/lazarus/0.9.28.2

This is the location of your Lazarus installation as mentioned above. Pleas check if the folder is correct, change if neccessary.

Save your changes an close the editor.

Type: *make all*

This should compile mocca, create a binary in the same folder and copy the mocca binary to your emc2/bin folder.

If you have Problems:

„Can:t find unit interfaces used by mocca“
This means that your installed LAZARUS does not have the compiled units for the Gtk2 widgetset.
Please Refer to the Lazarus documentation how to compile Lazarus and the Gtk2 Widgetset.

„Undefined Reference to“
This means that the Linker did not find a function or declaration in the linked files.
If you get this message, please check if you are using the matching EMC2 Version for the Makefile and check if the EMCDIR in the Makefile points to the right folder.

„Undefined Reference to ___DSO_HANDLE“
This is a problem of the gcc compiler on some computers. If you get this messsage change the following #define in your Makefile:

-UDSOHANDLE to -DDSOHANDLE

„Error___DSO_HANDLE was declared...“
This is a problem of the gcc compiler on some computers. If you get this messsage change the following #define in your Makefile:

-DDSOHANDLE to -UDSOHANDLE

If mocca was compiled successfully:

Take a look at your emc2 bin folder:

cd ~/emc2-2.4.1/bin

There is a file „mocca“ this is the mocca binary.

Next steps:

Edit your config- file (lets take axis_mm.ini)

cd ~/emc2-2.4.1/configs/sim

gedit axis_mm.ini

go to the [DISPLAY] section,
Change

DISPLAY = axis to *DISPLAY = mocca*

Add a new Section in the ini-file:

[MOCCA]
CONFIG= /home/my_name/moccagui/skins/default

CONFIG is the folder where your Layout and Config-File for mocca are located.

Save your config- file as *mocca.ini*

Go to your emc2/scripts folder

cd ~/emc2-2.4.1/scripts

type
./emc

Select */sim/mocca.ini* as the ini-file to start emc2.

Check the messages in the terminal.

Now Emc2 starts with mocca as display.

How to change the Design of mocca

The following files are read by mocca at start time:

config.xml

The config.xml file is the main configuration file for mocca.

The section that defines the OpenGl Colors used by mocca.

```
 <glcolors>
   <item name="bg" r="1" g="1" b="1"/>
   <item name="table" r="0" g="0" b="0.9" a="0.1"/>
   <item name="limits" r="0.7" g="0.7" b="0.6"/>
   <item name="cone" r="1" g="0.5" b="0" a="0.5"/>
   <item name="traverse" r="0.5" g="0.5" b="0.5"/>
   <item name="feed" r="0" g="0" b="1"/>
   <item name="toolpath" r="0" g="0" b="0"/>
  </glcolors>
```

Want a black background???

"bg" defines the RGB- Colors for the Background.
Change them to

```
 <item name="bg" r="0" g="0" b="0"/>
```

and mocca will start with a black OpenGl background.

The a=“0.1“ is the transparency value.
r,g,b define the R,G & B Colors.

Next section are the Button- Definitions for the Mocca-Buttons on the bottom of the main screen.
There are 10 Buttons available.

Example:

```
 <menujog>
   <item index="0" cmd="cmREF" bitmap="" text="Ref.-Fahrt..."/>
   <item index="1" cmd="cmTOUCHOFF" bitmap="" text="Antasten"/>
   <item index="2" cmd="cmTOUCHWIZ" bitmap="" text="Antasten..."/>
   <item index="3" cmd="cmTOOLS" bitmap="" text="Werkzg."/>
   <item index="4" cmd="cmSCRIPTS" bitmap="" text="Scripts..."/>
   <item index="5" cmd="cmCOORDROT" bitmap="" text="Ausrichten..."/>
   <item index="9" cmd="cmCLOSE" bitmap="" text="Beenden"/>
  </menujog>
```

<menujog> means that the following items are the items displayed in Manual (Jog) mode.

<item index="0" cmd="cmREF" bitmap="" text="Home..."/>

„Index“ is the Index of the Button, the range for Index is from 0 to 9 (10 Mocca-Buttons)
„cmd“ is the Command that will be executed by mocca when the user pressed the button.
Refer to the Mocca- commands list for further informations.
„bitmap“ is the name of a bitmap (at the moment only .png files) This image will be loaded at runtime from the folder CONFIG= /path_to_your_config/ in your mocca- inifile.
Example: bitmap="exit.png" will load the exit,png image for this button.
„text“ The caption of the button. Set to “” if your are using a bitmap.

Layout Files:

mocca uses xml- files to define the appearance of mocca. The xml- layouts allow the user to modify controls, add controls, change colors etc...

There is one main window. The Layout for the main window of mocca is the file mocca.xml.

Inside the main window are 2 areas for child windows:

First area is reserved for the simclient- window. The component name of this area is „PanelPreview“. The simclient window is the window that displays the OpenGl preview inside the component „PanelPreview“.

Second area displays the childs according to the emc2 mode that is active. The component name of this area is „PanelMaster“.

in Manual (jog) mode the PanelMaster displays the jog window.
In Auto (Run) mode the PanelMaster displays the run window and in
MDI mode the mdi window is displayed in the Panelmaster.

The main-buttons (those are located at the bottom of the main window) change according to the emc- mode (Manual,Auot,MDI)

Lets take a look at mocca.xml:

each component is described in the mocca.xml- file:
Do NOT change the name or class of a component!
If you want to modify something only edit the property of a component. Never change the class name or component name cause this will raise an error at startup.

```
<component name="LedMachineOn" class="TMocLed">
    <properties>
      <integer name="Left" value="880"/>
      <integer name="Height" value="14"/>
```

```
        <integer name="Top" value="152"/>
        <integer name="Width" value="80"/>
        <ident name="LedColor" value="$FF00"/>
        <boolean name="IsOn" value="false"/>
        <ident name="Color" value="clSilver"/>
      </properties>
    </component>
```

As you can see the compnent itself is described as normal Text.
The class of the component is a TMocLed. The name of the component is „LedMachineOn“. The name reflects the function of the Led. „Machine On“ means that this Led shows Emc2s Machine On Signal.

A Led control has 2 own properties:

LedColor: This is the color of the Led when the Led is ON.
IsOn: The initial state of the Led.

Colors can be defined by a String

starting with $ and the Hexadecimal notation of the R,G & B Colors,
$00FF00 means pure Green

or

a string strating with cl and the Name of the Color.
clRed means RED and clBlue means Blue. Available colors see: COLOR DEFINITIONS.

The other Items describe the postion and size of the component in screen pixels.
Each component has at least the LEFT,TOP,WIDTH and HEIGHT properties.

The value is always an integer.

WIDTH and *HEIGTH* may not be smaller than 1 Pixel.

Setting a Background image for the mainform (layoutfile: mocca.xml)

open the mocca.xml file and add the marked line:

```
<CONFIG>
  <Component>
    <component name="MainForm" class="TMainForm">
      <properties>
        ...
        <string name="Image" value="custombackground.jpg"/>

      </properties>
```

Mocca will load this image (custombackground.jpg) from the same configdir where the mocca.xml is located.

Mocca mainwindow will be sized to the size of the image. So be careful when you add a custom background image cause controls that are outside the mainwindow will not be visible.

Mocca commands list:

These are the mocca commands that may be used on buttons:

cmABORT: raises an emc_abort, stops any motion and puts the machine in the OFF status.
cmESTOP: stops any motion, turns Machine OFF and puts the Estop in the ON status
cmMACHINE: toggles Machine status between OFF and ON.
cmJOG: set emc mode to manual mode
cmAUTO: set emc mode to auto mode
cmMDI: set emc mode to MDI mode
cmSPMINUS: decrease spinlde speed
cmSPPLUS: increase spindle speed
cmSPCW: turn on spindle clockwise
cmSPCCW: turn on spindle counterclockwise
cmSPBRAKE: toggle spindle brake
cmFLOOD: toggle flood ON/OFF
cmMIST: toggle mist ON/OFF
cmREF: switch to submenu <menuref> in jog mode
cmREFX: home axis x
cmREFY: home axis y
cmREFZ: home axis z
cmREFA: home axis a
cmREFB: home axis b
cmREFC: home axis c
cmUNREF: unhome selected axis in jog mode
cmREFALL: home all axes
cmUNREFALL: unhoma all axes (not implemented yet)
cmTOOLS: switch to submenu <menutool> in jog mode
cmTOOLEDT: show the tool edit dialog
cmTOOLCHG: show the select tool dialog and execute a toolchange
cmTOOLCL: tool clamp ON (not implemented yet)
cmTOOLUNCL: tool clamp OFF (not implemented yet)
cmTOOLPAR: show the toolparam dialog (cutting speed limiter, not implemented)
cmTOOLMSR: measure tool (not implemented yet)
cmTOUCHOFF: switch to submenu <menutouch> in jog mode
cmTOUCHX: touchoff axis x
cmTOUCHY: touchoff axis y
cmTOUCHZ: touchoff axis z
cmTOUCHA: touchoff axis a
cmTOUCHB: touchoff axis b
cmTOUCHC: touchoff axis c
cmZEROALL: set all coords to 0
cmOFFSDLG: show the offsets edit dialog
cmTOUCHWIZ: show the touchoff wizard dialog

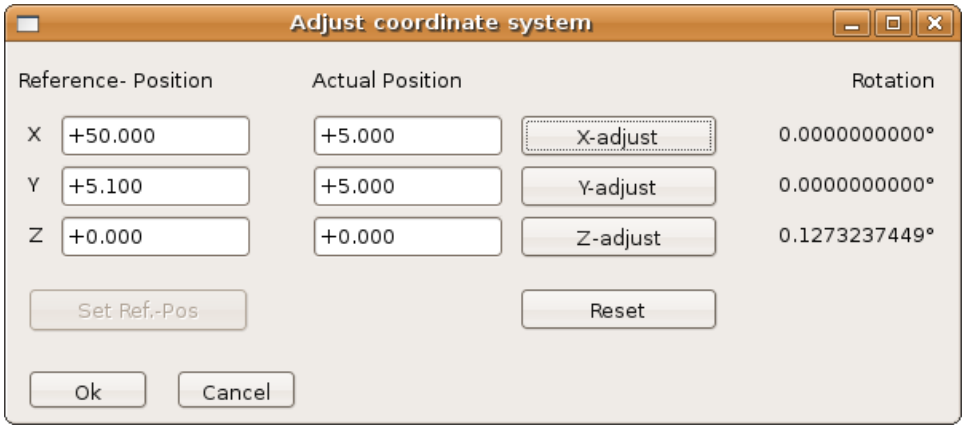
cmSETUP: show the setup dialog (not implemented yet)
cmLIMITS: toggle override limits if on limit switch
cmOPEN: open a file
cmRUN: execute a file from the first line
cmRUNLINE: execute a file from the selected line (in run mode)
cmSTOP: stop execution
cmSTEP: execute the follwing line of nc code
cmPAUSE: pause execution
cmOPTSTOP: toggle optional stops ON/OFF
cmBLOCKDEL: toggle blockdelete ON/OFF
cmRELOAD: reload the current file
cmMDIEXEC: execute the line entered in the mdi- edit field
cmMDIHIST: clear mdi history
cmEDITOR: open the current file in the editor
cmINCRDN: set the jog increment one down (First is continous)
cmINCRUP: set the jog increment one up
cmSHIFTUP: internal command, do not use!
cmSHIFTDN: internal command, do not use!
cmCOORDROT: show the coordinate system rotate dialog
cmUNITS: toggle between units mm / inches
cmCLOSE: close mocca
cmBACK: switch any submenu back to the normal menu
cmSCRIPTBASE: internal command, do not use!
cmSCRIPTS: switch to the submenu <scripts>
cmFEEDRESET: reset feed override to 100 %

mocca and HAL

mocca exports the following HAL pins:

moc.rotation_x float → out
moc.rotation_y float → out
moc.rotation_z float → out

the rotation_* pins are the rotation in degrees for the coordinate system.
The following dialog is the coordrotate dialog (cmCOORDROT)

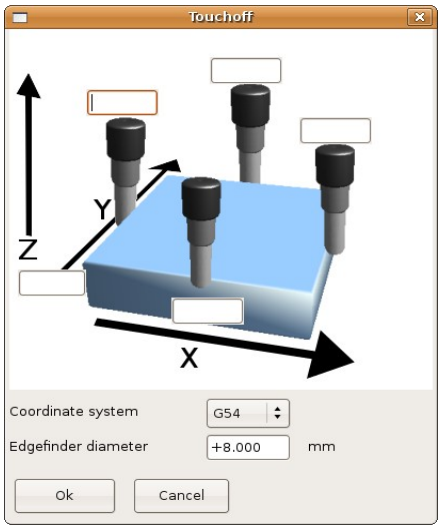


Example: the part is clamped on the machine table. First jog to the right bottom corner of your part. Press the <coordinate adjust> button. The current position is the reference position. Press the <Set Ref.-Pos> button and close the dialog with <ok>
Next jog to the left bottom corner of your part. Press the <coordinate adjust> button again. As you can see in the dialog above the left bottom Y coordinate differs 0.1 from the right bottom Y coordinate. Press the <X-Adjust> button. Now the z-rotation will show the angle in degrees that the coordinate system has to be rotated.

In your Hal pass this HAL pin (rotation_z) to your kinematic function to calculate the correct position.

Each time the adjust coordinate dialog is show the rotation_* values are set to 0.

Touchoff Wiz. Dialog



This ist the Touchoff Wiz Dialog.

There are 5 edit controls, indicating the position of your edgefinder.

Left of your part on X axis,
Right of your part on X axis
in front of your part on Y axis
in the back of your part on Y axis
and finally the top of your part on Z axis

The combobox- control shows the coordinate system.

The bottom edit- controls is the edgefinder diameter.
Preset value for the edgefinder diameter can be set in the config.xml:

```
<global>
  <float name="Edgefinderdia" value="10"/>
</global>
```