COSI-Measure v1.0

Software instructions

Revision History

Rev.	**Date** **(YYYY-MM-DD)**	**Description of Change**	**Author/** **Contributors**
1.0	2019-01-31	Initial version	Haopeng Han Lukas Winter
1.1	2019-06-19	Revision	Lukas Winter

Notes:

-

ABOUT

Please find below the software documentation for COSI measure a 3D multipurpose measurement system. If you find any flaws or if you have any questions/suggestions with regards to this document or project please let us know lukas.winter@ptb.de. Improving the quality of this work and its documentation makes it easier for others to reproduce and build upon this work.

If you find COSI measure useful in your work, please cite this paper:

Han H, Moritz R, Oberacker E, Waiczies H, Niendorf T and Winter L, „Open Source 3D Multipurpose Measurement System with Submillimetre Fidelity and First Application in Magnetic Resonance“, Scientific Reports, 7:13452, 2017

Introduction

This document describes the software of COSI Measure.

A more detailed description of the complete system can be found here:

http://www.opensourceimaging.org/project/cosi-measure/

and here:

Han H, Moritz R, Oberacker E, Waiczies H, Niendorf T and Winter L, „Open Source 3D Multipurpose Measurement System with Submillimetre Fidelity and First Application in Magnetic Resonance“, Scientific Reports, 7:13452, 2017 open source imaging .org

1. **Beagle Bone Black (BBB)**

Here are the instructions on how to set up the BBB: http://beagleboard.org/getting-started

1.1. Talk to the BBB

You can use the BBB stand alone (monitor over HDMI and mouse/keyboard over USB) or communicate (OS independent, Linux and Windows should work) e.g. with an SSH client, an ftp client, a remote desktop viewer or a DHCP server. You'll need to either connect a Cat.5 cable to the Ethernet interface of the BBB or optionally by using the mini-A USB port to your computer's standard USB port in order to use it as a serial port. Under Microsoft Windows you can use tools such as:

- putty
- DHCP server
- WinSCP
- VNC viewer

In Linux you don't need to install anything as these services are included in most Linux distributions (e.g. Ubuntu). E.g. you can use minicom to talk to the serial port. Open a shell and type "sudo minicom -s" to set the interface to "/dev/ttyACM0, 115200, 8n1, no flow control". Then you should be able to communicate with the BBB. You can use "ifconfig" to check the IP address of the Ethernet port and "arp-scan" to get the IP address of the BBB (provided that the DHCP service is running, if not, enable it). Now you can use ssh and sftp to login the BBB. When logging in as "username: debian, password: debian", you can also use the following command to start VNC on the BBB and view the desktop on your computer.

"x11vnc -bg -o %HOME/.x11vnc.log.%VNCDISPLAY -auth /var/run/lightdm/root/:0 –forever"

In our final setup, we use the BBB as a standalone device with a monitor, mouse and keyboard.

1.2. Installing/Booting OS

The BBB comes with a preinstalled OS on the on board eMMC. You can run another linux version either directly from the microSD card slot or flash the 4G on board memory. The latest images can be downloaded here: https://beagleboard.org/latest-images

and the SD card flashing instructions can be found here: https://elinux.org/Beagleboard:BeagleBoneBlack_Debian#Flashing_eMMC

We have used the Debian 8.7 machinekit image, which can be found here:

https://rcn-ee.com/rootfs/bb.org/testing/2017-02-12/machinekit/

Machinekit is a modem port of LinuxCNC. Our current system is supported by Machinekit and the software can be used to control COSI Measure. However in order to allow for precise movements based on the hardware settings, more work software wise is needed to make use fully of LinuxCNC. In order to precisely control COSI Measure we have written our own GUI (see below).

1.3. System settings

Before running COSI Measure, a few system settings might need to be performed, depending on the motors and drivers used in your setup.

The most important parameters related to the motion control are in the file:
bebopr_r2.c

Here you can change the microstepping factor, maximum speed, acceleration etc. of our current setup.

Important: Before running the system it is recommended to check each axis direction and functioning of the limit switches. For that it is best to use a slow maximum speed, a position of the probe somewhere around the center in X, Y and Z and manually testing the limit switches by starting the homing operation (see 1.4) while using some type of conductor that triggers the limit switches. After successful operation, the system settings can be adjusted to a higher speed and acceleration. E.g. our application of field mapping does not require high speed operation (the system can go much faster) so our limits are: x_axis 1000, y_axis 1000, z_axis 1000 and for acceleration we used x_axis 0.1, y_axis 0.1, z_axis 0.1.

<u>From microsteps to millimeters</u>
For our motors the step angle is 1.8°, so by default the steps per revolution are 200. When you set your motor drivers to 400 steps per revolution, the micro step factor is 2. You can adjust your microstepping factor via the stepper motor drivers (check also the COSI-Measure electrical assembly document). We have used 1600 pulse/rev leading to a microstepping factor of 8.

```
/*
 *  On systems with Pololu modules and DIP switches, the
 *  values here should reflect the DIP switch settings.
 *  If using a PEPPER board, the values below are used to
 *  configure the stepper drivers.
 */
unsigned int config_get_micro_step_factor( axis_e axis)
{
  switch (axis) {
  case x_axis:  return 8;
  case y_axis:  return 8;
  case z_axis:  return 8;
  case e_axis:  return 8;
  default:      return 8;
  }
}
```

The spindle used in our system leads to a linear motion of 5mm after a full rotation of the stepper motor. This can be configured (and changed if a different spindle is being used) here:

```
   *  PRUSA
   *
   *  X: 1:8  stepping, 0.9' motor, 16t pulley @ 3mm pitch => (16x3)/(8*360/0.9) => 0.015 mm
   *  Y: 1:8  stepping, 0.9' motor, 8t pulley @ 5mm pitch => (8x5)/(8*360/0.9) => 0.0125 mm
   *  Z: 1:32 stepping, 1.8' motor, 1:1 reduction @ 1.25mm /rev => (1.25)/(32*360/1.8) => 0.0001
   *  E: 1:8  stepping, 1.8' motor, 11:39 reduction @ 19mm /rev => (11/39*19)/(8*360/1.8) => 0.0
   */
   case x_axis:  return (5.0E-3) / (360 * config_get_micro_step_factor( axis) / 1.8);
   case y_axis:  return (5.0E-3) / (360 * config_get_micro_step_factor( axis) / 1.8);
   case z_axis:  return (5.0E-3) / (360 * config_get_micro_step_factor( axis) / 1.8);
   case e_axis:  return (5.0E-3) / (360 * config_get_micro_step_factor( axis) / 1.8);
#endif
   default:  return 0.0;
   }
}
```

<u>Axis direction</u>

The direction of the axis or the definition of the coordinate system can be changed here:

```
/*
 *  Specifiy the axes that need a reversed stepper direction signal
 */
int config_reverse_axis( axis_e axis)
{
  switch (axis) {
  case x_axis:  return 1;
  case y_axis:  return 1;
  case z_axis:  return 0;
  case e_axis:  return 1;
  default:  return 0;
  }
}
```

<u>Speed and acceleration</u>

```
/*
 *  Specify maximum allowed feed for each axis in [mm/min]
 */
double config_get_max_feed( axis_e axis)
{
  switch (axis) {
  case x_axis:  return 300.0;   // 0.00625 mm/step @ 60 kHz 22500.0; ba2200
  case y_axis:  return 300.0;   // 0.00625 mm/step @ 53 kHz 16000.0;
  case z_axis:  return 300.0; // 0.00039 mm/step @ 13 kHz
  case e_axis:  return  3000.0; // 0.00198 mm/step @ 25 kHz
  default:  return 0.0;
  }
}

/*
 *  Specify maximum acceleration for each axis in [m/s^2]
 */
double config_get_max_accel( axis_e axis)
{
  switch (axis) {
  case x_axis:  return 0.01;//0.1
  case y_axis:  return 0.01;
  case z_axis:  return 0.01;
  case e_axis:  return 1.0;
  default:  return 0.0;
  }
}
```

1.4. Running the system (without GUI)

After the configurations have been done, you have to compile the code. For that type:

source setenv

make clean
make

After the compiler finished you can start COSI measure (without GUI) by typing:
./mendel.elf

If you want to move the system you can use G-code. A documentation of the implemented G-code commands can be found in the comments of the file gcode_process.c

Here is a quick summary:

<u>Homing</u>
Before operation you'll need to set your reference 0,0,0 point by homing the system.

Homing of single axis:
G161x → (home negative, x-axis, you might need to type this command twice until the limit switches are reached)

After all axis have been homed, you can start moving the system to a dedicated position.

<u>Movement</u>
Moving the system (along X Z Z) can be performed by typing commands such as:
G1 X90.6 Y13.8 Z350

1.5. GUI
(coming soon)

Comments

-