3DHex 1.0.0 GUIDE

WARNING!!!!!
DO NOT leave your 3D Printer unattended, this is a very first beta version of the firmware..!!!

- Introduction
- Configuration
- USB printing
- SD Card printing

Image created with DesignEvo

INTRODUCTION

3DHex firmware is an open source project under GPLv3 license which controls the printing process of a 3D printer. The main purpose of the firmware is to use the CPU power of a typical desktop computer to generate a binary file containing the sequence of 0s and 1s that the microcontroller has to push out in order to print. The result binary can be saved for later SD Card printing or either streamed from the USB port to the microcontroller.

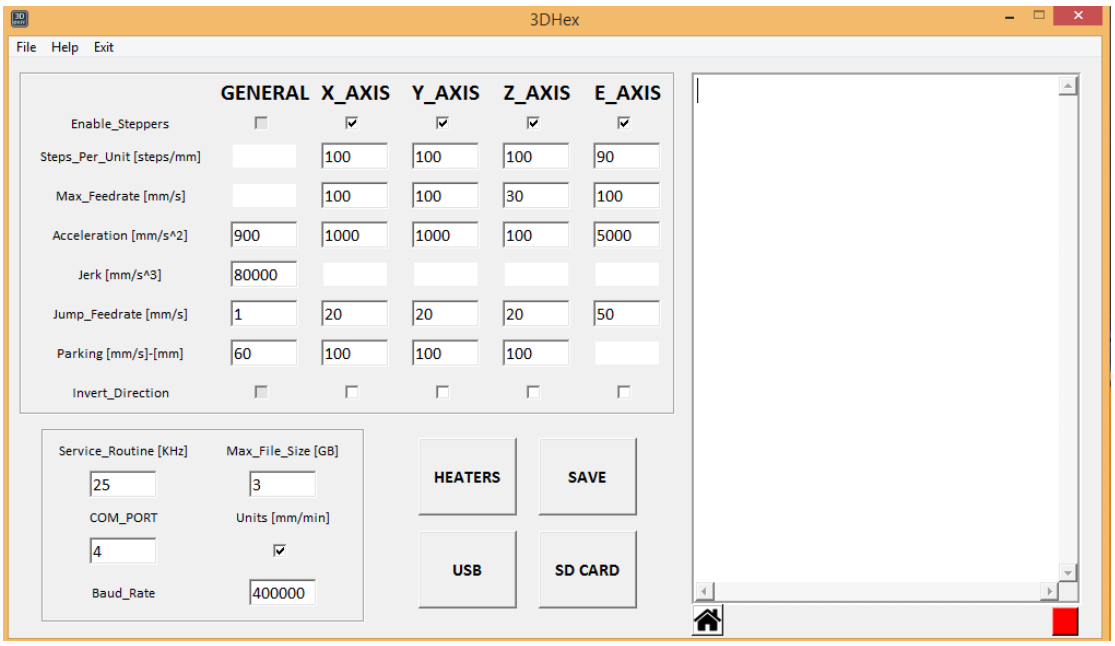
Also the firmware has constructed with a GUI interface which make the configuration process user friendly.

Features of 1.0.0 version:

- Jerk free S-curve profile
- Switching option between S-curve or trapezoid profile
- Real G02/G03 arc movement
- User friendly configuration
- Jump velocity (no acceleration)
- USB printing (15KHz step rate)
- SD Card printing (25KHz step rate)
- Basic thermal protection, maximum temp or broken thermistor
- Temperature control over interface only (M commands not supported)
- Only thermostat type temperature control (PID not supported yet)
- Tested only with RAMPS shield and 16x2 LCD
- Supported Gcode commands G90/G01/G1/G03/G3/G02/G2 (others will be ignored)
- Only absolute coordinates
- Only Cartesian printers
- Only one extruder supported
- Only min endstops homing through interface (G28 will be ignored)

CONFIGURATION

Main window settings



	GENERAL	X_AXIS	Y_AXIS	Z_AXIS	E_AXIS
Enable_Steppers	☐	✓	✓	✓	✓
Steps_Per_Unit [steps/mm]		100	100	100	90
Max_Feedrate [mm/s]		100	100	30	100
Acceleration [mm/s^2]	900	1000	1000	100	5000
Jerk [mm/s^3]	80000				
Jump_Feedrate [mm/s]	1	20	20	20	50
Parking [mm/s]-[mm]	60	100	100	100	
Invert_Direction	☐	☐	☐	☐	☐

Service_Routine [KHz]	Max_File_Size [GB]
25	3
COM_PORT	Units [mm/min]
4	✓
Baud_Rate	400000

Enable_Steppers: Enable or disable a stepper motor. When the process will end the stepper will be enabled again.

Steps_per_Unit: The steps per millimeter precision for each axis.

Max_Feedrate: The velocity limit of each axis.

Acceleration: The constant acceleration + the constant acceleration limits for each axis.

Jerk: The rate change of acceleration for S-Curve motion. If set to 0 the algorithm will switch to trapezoid motion profile.

Jump_Feedrate: The starting velocity of motion velocity profile + limits for each axis.

Parking: The velocity for moving to Parking + absolute coordinates.

Invert_direction: Inverts the positive move direction of an axis.

Service_Routine: The microcontroller's frequency for output toggling function. The maximum produced step rate calculated dividing this frequency by the factor 2.

Max_file_size: The maximum generated binary file size when printing from SD Card. Above this size and the printing process will be divided to parts.

COM_PORT: The serial port which the microcontroller is connected in order to communicate when printing from USB.

Units: The FEEDRATE units of the Gcode file.

Baud_Rate: The serial baud rate between desktop and the microcontroller. In order to change this you have to also change the MCU code.

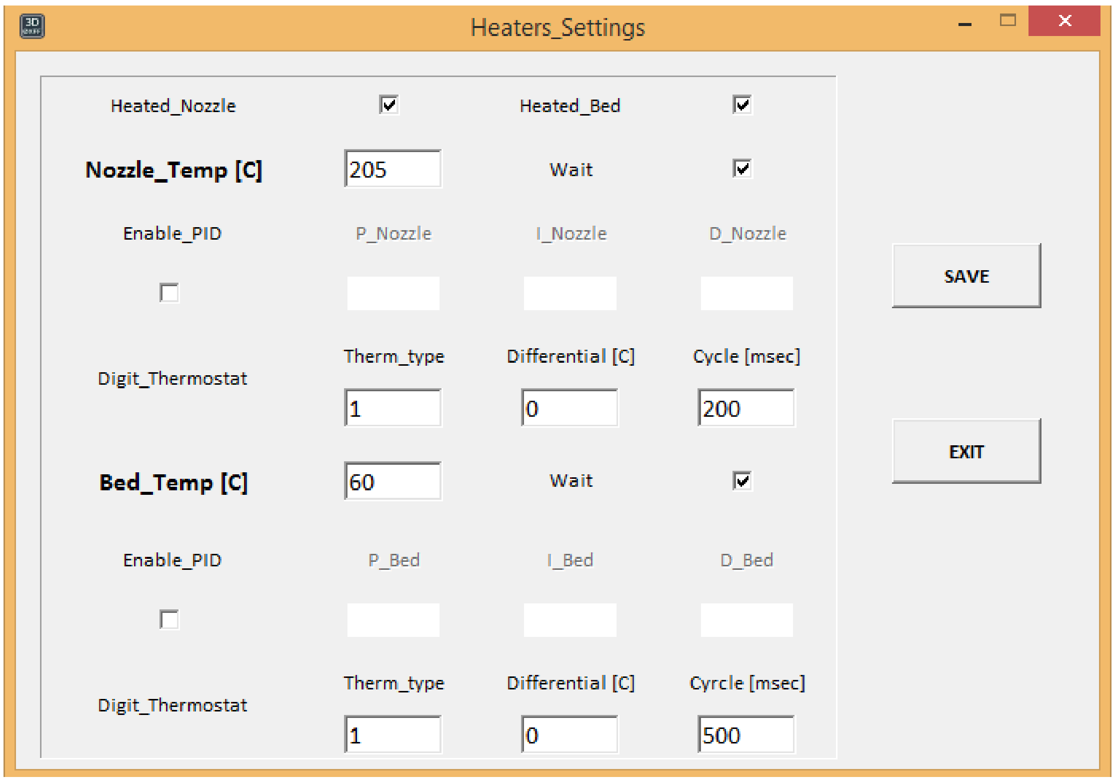
HEATERS: When this button is pressed it opens the bed and nozzle temperature settings.

SAVE: This button will save all the settings in the main window and the imported G code file for printing. Before every new G code commands you should save first.

USB: Starts the printing process from USB.

SD CARD: Opens a window in order to choose the location in which the generated binary file will be saved.

Heaters window settings



Heated_Nozzle	✓	Heated_Bed	✓
Nozzle_Temp [C]	205	Wait	✓
Enable_PID	P_Nozzle	I_Nozzle	D_Nozzle
☐			
Digit_Thermostat	Therm_type	Differential [C]	Cycle [msec]
	1	0	200
Bed_Temp [C]	60	Wait	✓
Enable_PID	P_Bed	I_Bed	D_Bed
☐			
Digit_Thermostat	Therm_type	Differential [C]	Cyrcle [msec]
	1	0	500

Heated_Bed // Heated_Nozzle: Enables or disables heated bed or heated nozzle at the start of printing process.

Nozzle_Temp // Bed_Temp: The target temperature each.

Wait: For each case when the wait check is selected the machine will first rise to the target temperature and then it will start to move. If both are selected then the machine will first heat up the bed and afterwards the nozzle.

Enable_PID: Enables the PID temperature controller (version 1.0.0 doesn't support this feature) by default it uses the thermostat type temperature control.

Differential: It is a thermostat feature not implemented yet.

Cycle: The time between each temperature read in the temperature control function for each PID and thermostat controllers.

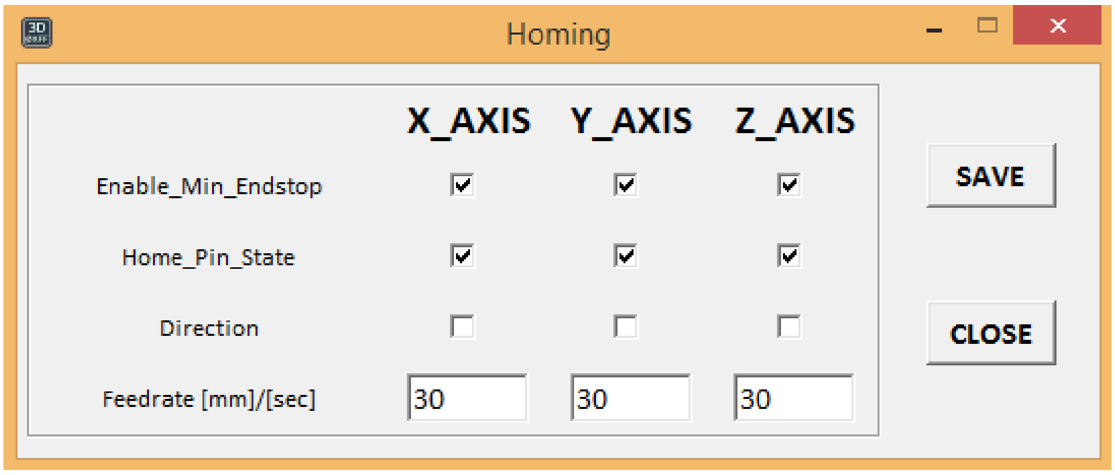
Therm_type: Defines the type of the thermistor from the table below.

value	Thermistor
0	Not used
1	EPCOS 100k(4.7k pullup)
2	Semitec 204GT-2(4.7k pullup)
3	Mendel-parts (4.7k pullup)
4	10k thermistor
5	Semitec 104GT-2(4.7k pullup)
6	100k EPCOS(4.7k pullup)
7	100k Honeywell 135-104LAG-J01(4.7k pullup)
71	100k Honeywell 135-104LAF-J01(4.7k pullup)
8	100k Vishay NTCS0603E3104FXT(4.7k pullup)
9	100k AL03006-58.2K-97-G1(4.7k pullup)
10	100k 198-961(4.7k pullup)
11	100k beta 3950(4.7k pullup)
12	100k Vishay NTCS0603E3104FXT(4.7k pullup)
13	100k Hisens 3950
20	PT100
66	4.7M Dyze
70	100K Hephestos 2
75	NTC 100K MGB18-104F39050L32
51	100k EPCOS(1k pullup)
52	200k Semitec 204GT-2(1k pullup)
55	100k Semitec 104GT-2(1k pullup)
1047	Pt1000 with 4k7 pullup
1010	Pt1000 with 1k pullup
147	Pt100 with 4k7 pullup
110	Pt100 with 1k pullup

Save: Just saves the settings.

Exit: Closes heaters settings window. Warning if you don't pressed save first the new settings will be lost.

Homing window settings


Homing

	X_AXIS	Y_AXIS	Z_AXIS
Enable_Min_Endstop	✓	✓	✓
Home_Pin_State	✓	✓	✓
Direction	☐	☐	☐
Feedrate [mm]/[sec]	30	30	30

Enable_Min_Endstop: Enables or disables the homing process for each axis.

Home_Pin_State: Defines the endstop's state at homing position.

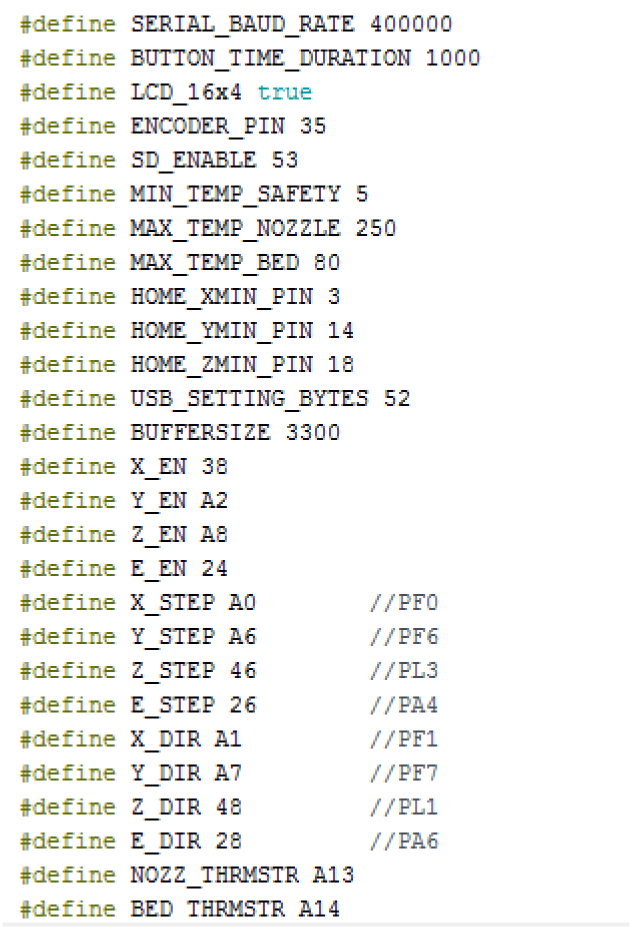
Direction: Defines the move direction for homing. Be careful wrong direction and the printer will crash at the other end of endstop.

Feedrate: The homing velocity for each axis. Be aware that during homing process no acceleration is applied to any motion.

SAVE: Will save current homing settings.

CLOSE: Will exit from homing settings window. Be careful to save first.

MCU controller parameters



```
#define SERIAL_BAUD_RATE 400000
#define BUTTON_TIME_DURATION 1000
#define LCD_16x4 true
#define ENCODER_PIN 35
#define SD_ENABLE 53
#define MIN_TEMP_SAFETY 5
#define MAX_TEMP_NOZZLE 250
#define MAX_TEMP_BED 80
#define HOME_XMIN_PIN 3
#define HOME_YMIN_PIN 14
#define HOME_ZMIN_PIN 18
#define USB_SETTING_BYTES 52
#define BUFFERSIZE 3300
#define X_EN 38
#define Y_EN A2
#define Z_EN A8
#define E_EN 24
#define X_STEP A0        //PF0
#define Y_STEP A6        //PF6
#define Z_STEP 46        //PL3
#define E_STEP 26        //PA4
#define X_DIR A1         //PF1
#define Y_DIR A7         //PF7
#define Z_DIR 48         //PL1
#define E_DIR 28         //PA6
#define NOZZ_THRMSTR A13
#define BED THRMSTR A14
```

SERIAL_BAUD_RATE: The baud rate of serial connection with the host when printing from USB. This value must be the same as in host.

BUTTON_TIME_DURATION: The time required in [msec] to press the rotary encoder button connected to ***ENCODER_PIN*** in order to start or terminate the printing process.

LCD_16X4: If true enables messages to LCD display. Only 16x2 size supported.

SD_ENBLE: The SD card wiring SS pin.

MIN_TEMP_SAFETY: Below this value the printer will terminate any process because of a broken or unconnected thermistor.

MAX_TEMP_NOZZLE // MAX_TEMP_BED: For any reason if read temperature above these limits any process will be terminated.

HOME_PINS: The pins that the endstops are wired with the MCU.

USB_SETTINGS_BYTES: The size in bytes of settings to receive from the host or read from the SD. Every time the printing process starts MCU has to read those values in order to work. **WARNING!!!!! DO NOT CHANGE THIS FOR ANY REASON!!!**

BUFFERSIZE: The size of buffer that contains all the raw binary for the outputs states. The MCU has two buffers, one is pushed out while the other filled with new data. The larger the buffers the better, this has a result to allocate almost all. **WARNING!!!!! DO NOT CHANGE THIS FOR ANY REASON!!!**

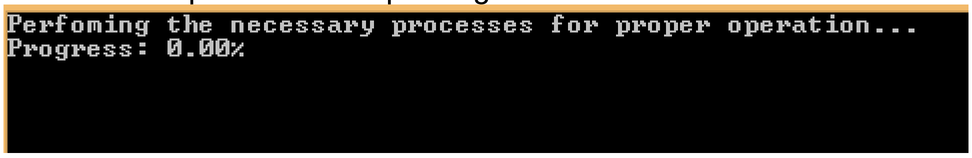
REMAINING PARAMETERS: Those parameters are just the pins that RAMPS shield is wired with the stepper motors drivers and the thermistors inputs. Be aware that the extruder driver works only at **E0** position.

USB PRINTING

Once you have done with configuration it's time for the first print. In order to print from the USB just plug in your 3D printer to one of the USB ports. It is recommended to connect the device with USB 3.0.

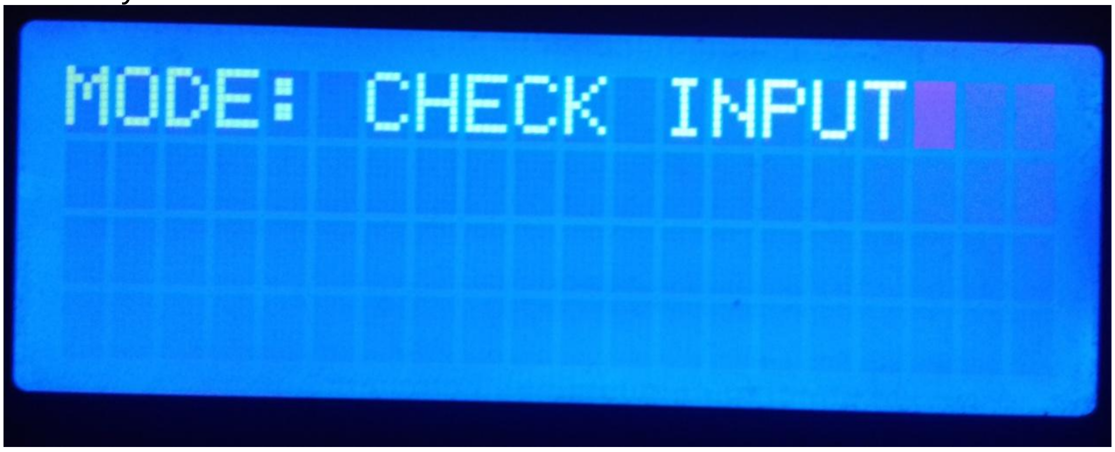
- Make sure which ***COM_PORT*** number you connected the printer in order to establish connection. You can find it from Arduino IDE.
- The ***Baud_Rate*** must be the same for both MCU and Host controller. The recommended and tested default value is 400000, the key is to have the highest possible baud rate without failure.
- It is recommended when printing from USB to set the ***Service_Routine 25 KHz*** or below. If this value set too high the printer will stall or terminate the process by itself.
- If the printer has endstops make sure to enable homing routine into the **homing** control panel window, click **SAVE** before CLOSE.
- Don't forget about the temperature settings for bed and nozzle into the **heaters** control panel window, click **SAVE** before EXIT.
- To select the file for printing just click on ***File*** >> ***Open*** >> ***Choose the Gcode file you want to print.*** The Gcode will appear into the text box.
- Figure out what the Feedrate ***Units*** of the Gcode commands and set the right one through the checkbox.
- Click **SAVE.**
- Click **USB.**

Into the command prompt will appear “Performing the necessary processes for proper operation…” It may take a while to execute the homing routine and heat up before start printing.

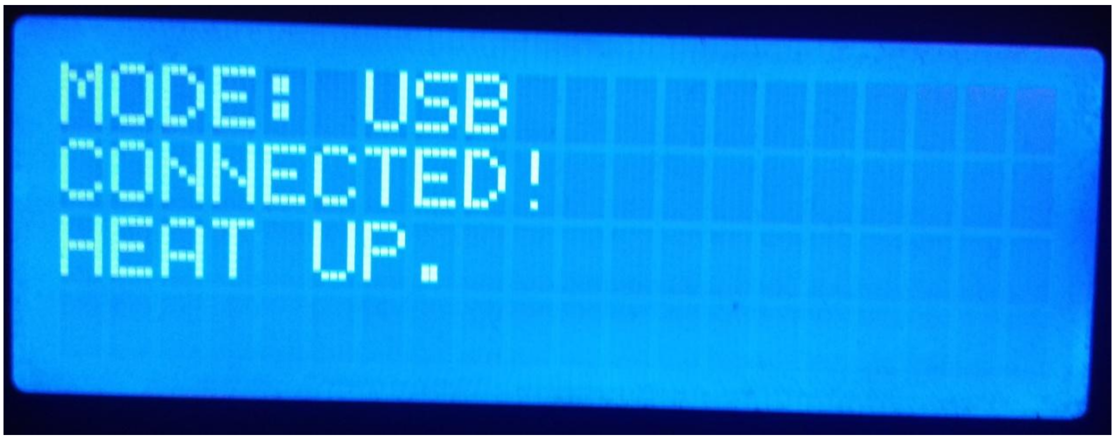


```
Perfoming the necessary processes for proper operation...
Progress: 0.00%
```

If you have connected an LCD 16x2 size then the LCD will show:

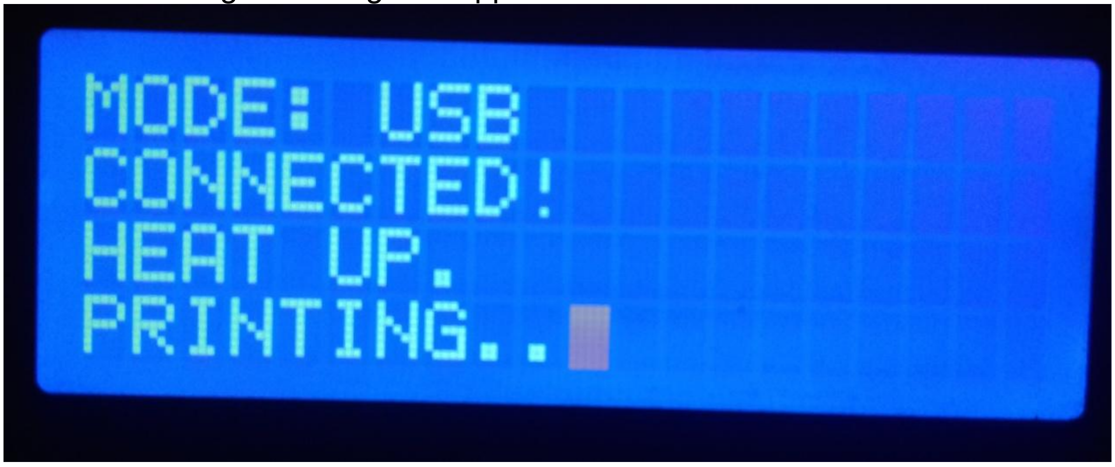


```
MODE: CHECK INPUT
```



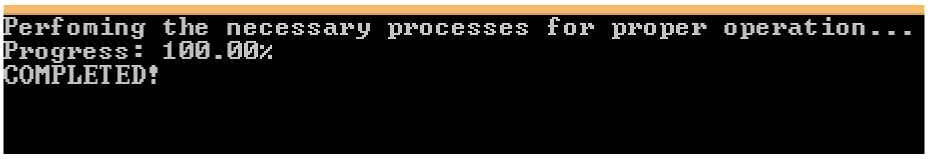
```
MODE: USB
CONNECTED!
HEAT UP.
```

During printing, the progress will be shown up on the cmd while in the LCD the message “Printing” will appear.



```
MODE: USB
CONNECTED!
HEAT UP.
PRINTING..
```

At the end of the printing “COMPLETED!” will printed to the cmd while “TERMINTED” on the LCD screen.



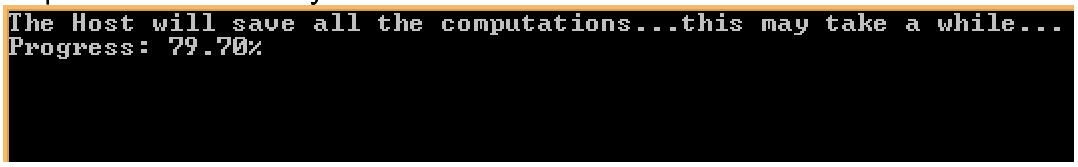
```
Perfoming the necessary processes for proper operation...
Progress: 100.00%
COMPLETED!
```

SD CARD PRINTING

Once you have done with configuration it’s time for the first print via SD Card. Printing from SD Card is more reliable since there is no real time critical communications. Also the SPI interface with the SD is faster than the USB to Serial resulting higher step rates.

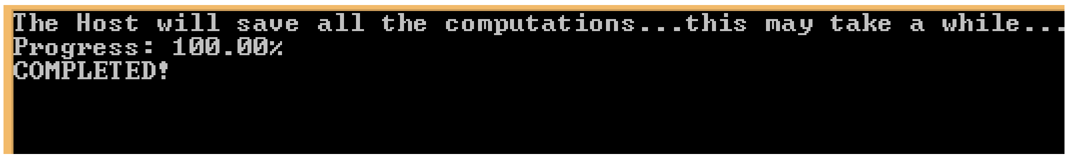
- It is recommended when printing with SD card the ***Service_Routine*** value to stay below ***45 KHz*** to prevent stalls and terminates.
- If the printer has endstops make sure to enable homing routine into the **homing** control panel window, click **SAVE** before CLOSE.
- Don’t forget about the temperature settings for bed and nozzle into the **heaters** control panel window, click **SAVE** before EXIT.
- Set value 3.5 GB or below for ***Max_File_size.***
- To select the file for printing just click on ***File*** >> ***Open*** >> ***Choose the Gcode file you want to print.*** The Gcode will appear into the text box.
- Figure out what the Feedrate ***Units*** of the Gcode commands and set the right one through the checkbox.
- Click **SAVE.**
- Click **SD CARD** and select where the binary file you want to be saved.

Into the command prompt will appear "The Host will save all the computations…this may take a while…"



```
The Host will save all the computations...this may take a while...
Progress: 79.70%
```

And when the binary file created all the way "COMPLETED!" will show up. **WARNING!!!** ***DO NOT RENAME THE BINARY FILE ELSE THE MCU WILL NOT BE ABLE TO OPEN IT..!!!***

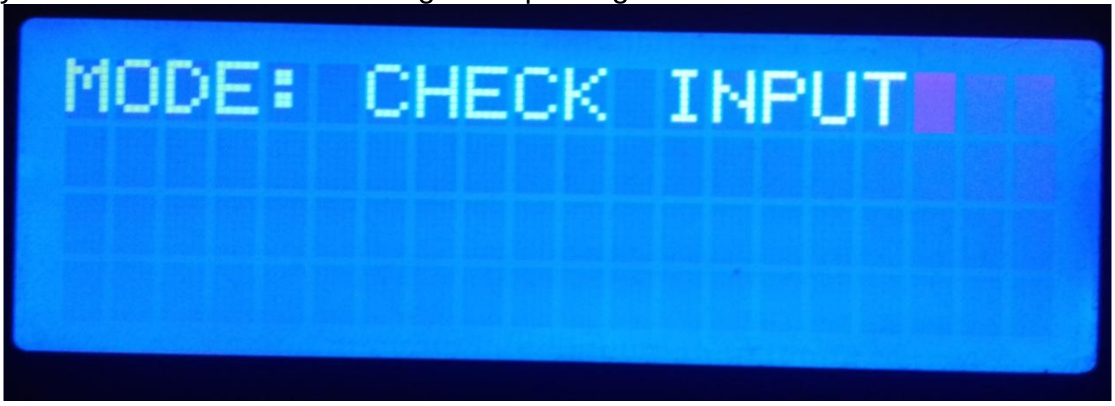


```
The Host will save all the computations...this may take a while...
Progress: 100.00%
COMPLETED!
```

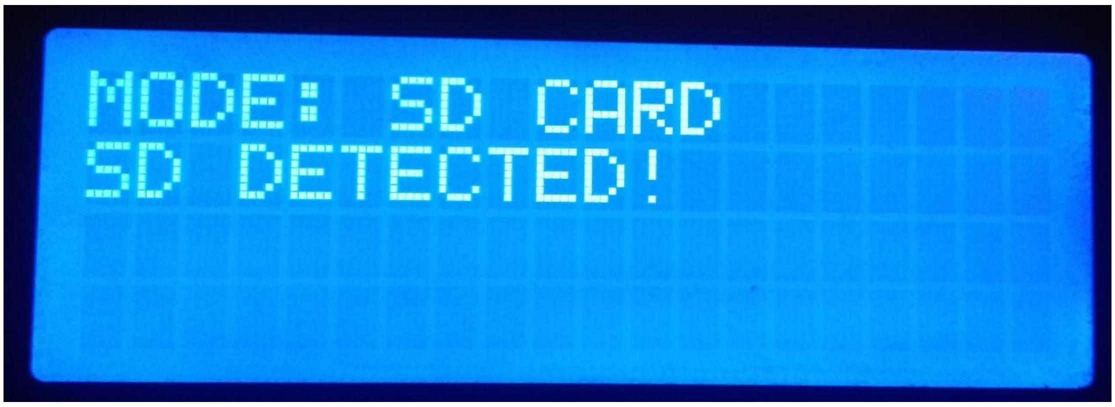
In order to use an SD Card first it has to be reformatted to FAT32. The maximum file size that supported by FAT32 is 4GB be careful with that. Once the binary was created you can re use it without wait again the program to generate it.

- Move the generated binary into the SD and place it on the machine.
- Press the encoder button about a second and the printing process will start automatically.

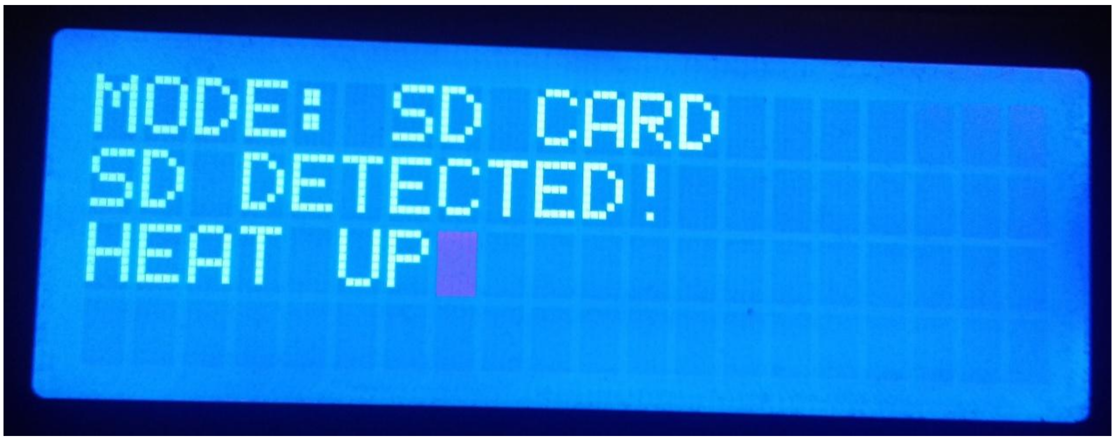
If an LCD 16x2 size is connected and enabled from MCU code then you will be able to see messages for printing state.



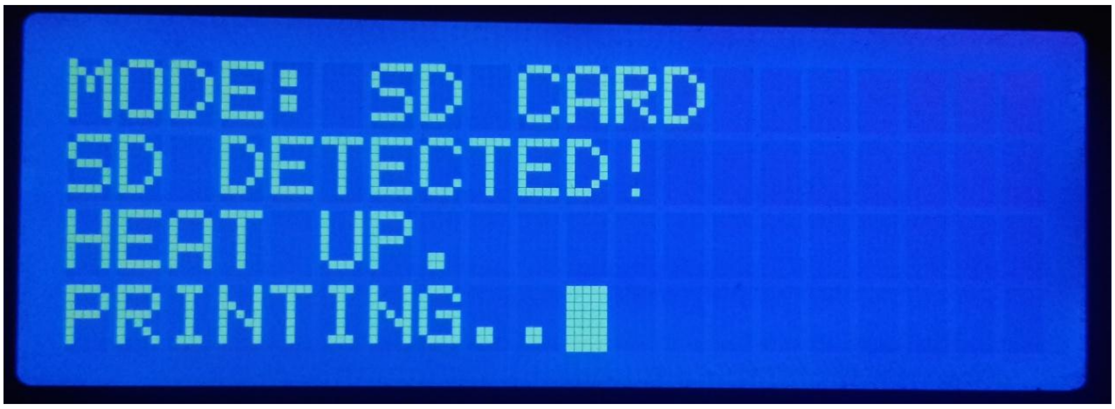
MODE: CHECK INPUT



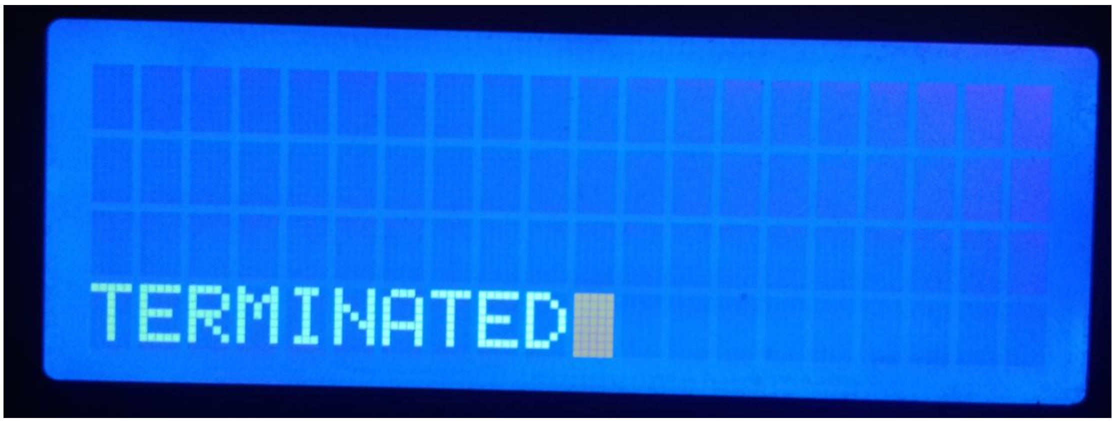
MODE: SD CARD
SD DETECTED!



```
MODE: SD CARD
SD DETECTED!
HEAT UP
```



```
MODE: SD CARD
SD DETECTED!
HEAT UP.
PRINTING..
```



```
TERMINATED
```