Voron 2.4 setup with Duet2wifi RRF3 and simple expansion board by Jantec.nl © 2020-10

Version 0.2 dated October 5th, 2020 Compiled by Jan Griffioen, Amsterdam, NL, Europe

Please donate $1 to my paypal account if you use (parts of) my config.g file so I can continue developing nice stuff for you to download

Content:

Inhoudsopgave

Overview

This config.g setup for Duet2 (either wifi or ethernet) has been compiled from several sources where I gathered bits and pieces, and I used my knowledge of my earlier build of a cartesian machine with a Duet2wifi, and my experiences of building my current 310/310/300 Voron with 2x SKR1.4 turbo and octopi.

Specifications

For this new to be built Voron 2.4 project the specifications are:

- Duet2wifi motherboard, 24Volt
- Voron 2.4 machine with a buildvolume of 510(w) x510(d) x550(h) mm
- Chinese 5x expansion board with 4 plug-in 2209 drivers
- Triple Bowden hotend with single nozzle
- Triple remote extruders
- DOTSTAR or WS2812 LEDs underneath the hotend to indicate the hotend heater status AND shine white when printing, for checking the printing quality during the job

Why DUET2

The Duet is chosen by me for the following reasons:

- I know the Duet2wifi board well, since I previously a.o. built a Cartesian printer with it
- The Duet2wifi integrated drivers work very well
- The Duet2wifi board is very stable
- The Duet2wifi makes it very easy to change things on the fly without the need to re-compile anything in the firmware
- The Duet2wifi has Integrated wifi and control software, This works very easy and better than octopi (and more controls are available like online adapting all files on the SD remotely)
- As add-on to the Duet2, much is available like PT100 add-on interface boards, dual driver add-on, quad-driver add-on, 5-port driver add-on, and the Duex versatile add-on board. This makes the Duet the most flexible board I ever encountered and therefore it is my choice to go with
- An octopi setup with an RPI4B and cams etcetera is always possible as add-on, by just connecting the Octopi to the USB connector of the Duet2Wifi.

Approach and preconditions

To have everything working properly with the Voron setup, a couple of files on the Duet's SD-card need to be changed like config.g, bed.g, homez.g, homeall.g and some macros need also be changed, like bed levelling and so on. This is all done via a PC/tablet/phone, via the Duet's web interface after you have logged in to this on the Duet2wifi.

For general use of a Duet2wifi board, please check the available user groups for this info.

Generally, the following updates need to be available on the Duet2wifi board:

- Reprap firmware 3.1 or later, due to the fact that commands and its use have changed dramatically versus firmware 2.X. My config and macro setup is fully based om FFR 3.1.
- Correct jumper settings for power etcetera

- Add-on 5V power board for the Arduino CPU and WS2812 LEDs, making use of the WS2812 LEDs to shine colored light from underneath the hotend, with dedicated colours per heating phase.

Changes to Config.g

This is the main setup file for the machine, the following config changes are made:

(full file is available at download CONFIG.G from my website)

Set the network to your wifi setup where you will use the Duet2wifi:

```
; Voron500 config file for dc42 Duet wifi firmware
; JanTec.nl 5-10-2020 for reprap V3.0

; Prologue and  comms section
M111 S0                         ; Debug off
M550 PVoron500                  ; Machine name (can be anything you like)
M551 Preprap                    ; Machine password
M552 S1                         ; WIFI ON

;*** If you have more than one Duet on your network, use different MAC addresses, so change the last digits
M540 P0xBE:0xEF:0xDE:0xAD:0xFE:0xBD ; MAC Address

;*** Adjust the IP address and gateway in the following 2 lines to suit your wifi network setup. The used addresses are my home router's settings 192.168.178.XXX, please use your own settings here!
M552 P192.168.178.33            ; IP address (0.0.0.0 = use DHCP)
M554 P192.168.178.1             ; Gateway
M553 P255.255.255.0             ; Netmask
M555 P2                         ; Set output to look like Marlin
M575 P1 B57600 S1               ; Comms parameters for PanelDue if you have one
```

Motor settings

Drive settings 1xAlpha (X),1x Beta (Y), 4xZ and 3xExtruder

```
; --- drive map ---
;   _______
;   | 6 | 7 |
;   | ----- |
;   | 5 | 8 |
;    -------
;     front

; Drive directions
M569 P0 S1 ; A
M569 P1 S1 ; B
M569 P3 S1 ; Extruder #1
M569 P4 S0 ; Extruder #2
M569 P5 S0 ; Z1
M569 P6 S1 ; Z2
```

M569 P7 S0 ; Z3
M569 P8 S1 ; Z4

- With the M584 command you map the motors with drivers. The Z drivers are located on the added expansion board that has plug-in 2209 drivers. You can either use the Chinese expansion board I used, or a Duex which is relly complete overkill, or you can use 2 pieces of the 'original' Duet 2-channel expansion boards that already include the driver IC's.
- With the M671 command you set the coordinates of each motor. For correct gantry levelling it is very important to set this exactly correct. The coordinates must match with the physical position of the motors. For example, the first Z motor is connected to plug number 5 (1st physical plug of the expansion board) of the expansion board (M584 Z5) and is the motor that is located on coordinates x = 560 (buildvolume+50mm) and y = 520 (build volume +10). The second motor is connected to the plug number 6 (2nd plug) of the expansion board and is physically located on coordinates x = -50 (always -50, regardless the build volume) and y = 520 (build volume+10).
- Finally with the M569 commands the direction of the motors are defined, please check this after the initial build and correct the directions if needed. If correcions are required you can either do this in config.g or physically by reversing the plug(s) that go(es) into the board(s).

Motor settings for microstepping, movement, jerk, acceleration and amps

```
; Motor mapping and steps per mm
M584 X0 Y1 Z5:6:7:8 E3:4
M350 X16 Y16 Z16 E16:16 I1      ; Use 1/16 microstepping with interpolation everywhere
;M92 X80 Y80 Z400               ; Set XYZ steps per mm (1.8deg motors)
M92 X160 Y160 Z800              ; Set XYZ steps per mm (0.9deg motors)
M92 E560:560                    ; Set Extruder steps per mm (Mobius 3)
M350 Z16 I0 ; disable Z interpolation
; Drive currents
M906 X1200 Y1200 Z1200 E1000    ; XYZ and E current
M906 I30                        ; Idle current percentage
M84 S120                        ; Idle timeout
; Accelerations and speed
M566 X900 Y900 Z60 E8000        ; Set maximum instantaneous speed changes (mm/min)
M203 X18000 Y18000 Z3000 E15000 ; Set maximum speeds (mm/min)
M201 X2000 Y2000 Z250 E1800     ; Set maximum accelerations (mm/s^2)
M204 P1500 T2000                ; Set printing acceleration and travel accelerations
```

- With M350 command you set the microstepping for the drivers, in my case I set all the movement motors (X, Y and Z) with 256 microsteps without interpolation and the extruder to 256 with interpolation.
- M92 command set the number of steps by mm for each movement (axis and extruder)
- M566 commando sets the jerk

- With M203 you set the maximum speed of the movements
- With M201 command you set the accelerations
- Finally, with M906 command you set the amperage for each motor, this is the basic setup, because in a further post I will explain how to use macros to change this values on the fly when you print.

Axis Límits

When you made the setup of the limits, you are defining the coordinate system for your printer, in my case, when the printer reach the X and Y endstops, the machine is at the maximum values (510 mm). By other hand when the FSR is touched the Z is at the minimum position

- With M208 commands you set the minimum and maximum limits for each axis.
- With M574 command, I said that when the endstop for X and Y axis are reached, it has the maximum value for these axis (510 mm), on the other hand, when the FSR is touched Z is at minimum value (unless this is not completely true, as you can see on the next post)

Mechanical Z-probe

```
; Select inductive probe
; P4:      connected to Zmin SIG and GND
; I0:      P4 expects NC, TL-Q5MC2-Z is also NC
; T18000:  Move to probe points at 300mm/s
; F1200:   Probing Speed: 20mm/s
; H5:      Dive height: 5mm
; A5 S0.01: Perform up to 5 touches until change is below 0.01mm
; B1:      Turn off heaters while probing
M400
M558 P8 I0 T18000 F1200 H5 A5 S0.01 B0 R0.2
G31 T8 P500 X0 Y25 Z3.45 ; inductive probe offset, not critical, only used for coarse homing
G4 P200
```

Bed leveling

```
; Bed levelling

M671 X-65:-65:365:365 Y-20:380:380:-20 S20      ; Define Z belts locations (Front_Left, Back_Left, Back_Right, Front_Right)

M557 X25:275 Y25:275 S25        ; Define bed mesh grid (inductive probe, positions include the Z offset!)
```

Triple Hotend

with PT100 thermocouple, 1 nozzle, 3 extruders and 3x Bowden setup

```
; Hotend #1 heater
M305 S"Hotend" P1 X200 W4                    ; 1st nozzle is 4-wire PT100, first channel
;M307 H1 A568.8 C203.2 D4.0 S1.00 V24.5 B0      ; E3D V6 + PT100 PID, 30W heater
;M307 H1 A365.9 C236.7 D4.9 S1.00 V24.5 B0      ; E3D Volcano + PT100 PID, 30W heater
;M307 H1 A614.3 C180.2 D5.3 S1.00 V24.4 B0      ; Mosquito + PT100 PID, 50W heater
M143 H1 S300                                 ; Set temperature limit for heater 1 to 300C
```

- On this part, I used a PT100 instead for the hotend temp sensor, with the original Duet interface board.
- Add/replace in Config.g: ; Thermistor section for the hotend

M308 S1 P"spi.cs1" Y"rtd-max31865" ; create sensor number 1 as a PT100 sensor in the first position ;on the Duet 2 daughter board connector, 2 wires, 1 on pin 2 and the other on pin 3. ;Connect pins 1 ;to 2 and 3 to 4 with a blob of solder first.

- With M305 commands you set the thermistors for the heated bed.
- With M143 you define the maximum temperature for the hotend and the heated bed.
- M307 is used to configure the PID for the hotend.
- You need to setup only the layer fans, the hotend fan is connected to a always on fan plug. The layer fans to fan connectors 0 and 2, because the connector 1 is dedicated to a thermostatic controlled fan.
- On the tools section you manage the tool itself, attaching to it the hotend and the layer fans with M563 command.

Bed heater

```
; Bed heater, dual thermistor setup (one for the heater + one for the bed)
M305 S"Bed Plate" P0 X0 R4700 T100000 B3950    ; Beta3950 stud thermistor on the edge of the plate
M307 H0 B1 S1                                ; 100% PWM, bang-bang mode
M305 S"Bed Heater" P103 X3 R4700 T100000 B3950  ; Beta3950 thermistor inside the Keenovo heater
M143 P100 H0 X103 A2 C0 S115                 ; make sure silicone heater stays below 115°C
M143 P101 H0 X103 A1 C0 S125                 ; make sure silicone heater shuts down at 125°C
M143 H0 S110                                 ; maximum bed temperature
```

Fans, tools and epilogue config.g

```
; Fans
;M106 P3 S1 I0 H1 T50           ; E3D V6 Hotend fan, turns on if temperature sensor 1 reaches 50 degrees
M106 P3 S0.6 I0 H1 T50          ; Mosquito Hotend fan @ 60%, turns on if temperature sensor 1 reaches 50 degrees
```

```
M106 P4 S0 I0 H-1                ; Part cooling fan, no thermostatic control
;M106 P5 T45:65 F50 H100:101:102     ; Electronics bay fan, turn on gradually if MCU is over 45C or any TMC driver is over temp
M106 P8 S1 H0 T50                ; Chamber filter fan, turn on when bed is hotter than 50C

; Tools
M563 P0 D0 H1 F4                 ; Define tool 0, use fan #4 for M106
G10 P0 X0 Y0 Z0                  ; Set tool 0 axis offsets
G10 P0 R0 S0                     ; Set initial tool 0 active and standby temperatures to 0C

; Pressure advance
M572 D0 S0.2

M501                             ; load config-override.g
T0                               ; select tool 0
```

MACROS

to setup the Z zero position by modifying homez.g and homeall.g files,

homez.g

```
; homez.g

; called to home the Z axis
M564 H0; Allow movements before homing
M98 P"/macros/print_scripts/z_current_low.g"
M98 P"/macros/print_scripts/speed_probing.g"
G91
G1 Z15 F2000; Lift Z relatively to current position
G90; Back to absolute positioning
M98 P"/macros/print_scripts/activate_z_probe.g"
M98 P"/macros/print_scripts/goto_bed_center.g"
G30 Z-9999; Coarse homing with the inductive probe
M98 P"/macros/print_scripts/z_current_high.g"
M98 P"/macros/print_scripts/speed_printing.g"; Restore high Z currents
; Note that homing Z does not set the final Z offset used for printing!
; You *must* probe Z with the Z switch before checking/calibrating the Z offset.
M564 S1 H1; Homing done, enforce limits
```

Homeall.g

```
; homeall.g

; called to home all axes
M564 H0; Allow movements before homing
G91; Relative positioning
M98 P"/macros/print_scripts/z_current_low.g"
G1 Z15 F2000; Lift Z
```

```
M98 P"/macros/print_scripts/z_current_high.g"
M98 P"/macros/print_scripts/xy_current_low.g"; Lower AB currents
G1 X600 Y600 F2400 S1; Coarse home X or Y
G1 X600 S1; Coarse home X
G1 Y600 S1; Coarse home Y
G1 X-5 Y-5 F9000; Move away from the endstops
G1 X600 F360 S1; Fine home X
G1 Y600 S1; Fine home Y
M98 P"/macros/print_scripts/xy_current_high.g"; Restore high AB currents


; Absolute positioning
G90
M98 P"/macros/print_scripts/z_current_low.g"
M98 P"/macros/print_scripts/activate_z_probe.g"
M98 P"/macros/print_scripts/goto_bed_center.g"
G30 Z-9999; Coarse homing with the inductive probe
; Note that homing Z does not set the final Z offset used for printing!
; You *must* probe Z with the Z switch before checking/calibrating the Z offset.
M98 P"/macros/print_scripts/z_current_high.g"; Restore high Z currents
M98 P"/macros/print_scripts/speed_printing.g"; Restore normal speed & accel
M564 S1 H1; Homing done, enforce limits
```

bed.g

```
;create the bed.g file to level the gantry

; bed.g
; called to perform automatic bed compensation via G32
M561; Clear any bed transform
G28; Home all axes
M98 P"/macros/print_scripts/activate_z_probe.g"; Activate the z probe and lower the z motor currents
M98 P"/macros/print_scripts/z_current_low.g"
M98 P"/macros/print_scripts/xy_current_low.g"; Lower AB currents
M98 P"/macros/print_scripts/speed_probing.g"; Setup low speed & accel
M98 P"/sys/bed_probe_points.g"; Probe the bed at 4 points
M98 P"/sys/bed_probe_points.g"; Repeat right away for more precision
M98 P"/sys/bed_probe_points.g"; Repeat right away for more precision
M98 P"/macros/print_scripts/do_z_switch_probe.g"; Final Z height adjust
;M98 P"/macros/print_scripts/goto_bed_center.g"
M98 P"/macros/print_scripts/speed_printing.g"; Restore normal speed & accel
M98 P"/macros/print_scripts/xy_current_high.g"; Restore high current and make sure Z probe is active
M98 P"/macros/print_scripts/z_current_high.g"
G29 S1; Load previously probed bed mesh
```

Dotstar or WS2812 LEDs

There are 2 working options to get controlled LED's connected to your Duet2:

WS2812 LED(s)

To connect WS2812 LED's to a Duet, everything needs to be made since nothing is available. The reason is, that the way these 2812 LED's work, they need constant steering from a processor. This will take too much attention of the Duet's CPU. However, an implementation is possible with an add-on Arduino processor to control the WS2812 LED(s) and the Arduino can easlily communicate with the Duet by using a macro on the Duet to tell the Arduino what to do.

The absolute TOP is that Arduino can steer a lot of LED's, and can do so in multiple strings. So you could make LED's or Led strips anywhere in, on, at, and under your machine and control everything individually, or let it be controlled by the printing process. How awesome would it be to have LED's vertically that grow in height and color as your printing grows…

My knowledge of Arduino and WS2812 should do to get this to work. I might also make a translation from Dotstar protocol that comes from the Duet2 and use that as input for the Arduino to control the WS2812 LEDs. But, in my view it would be better to get the Arduino controlled by a (couple of) macros that control specific pins on the Duet that connect to the Arduino. That would give much more flexibility and could be used to drive a lot of WS2812 LEDs, if required.

Also see for a working implementation of this: https://www.youtube.com/watch?v=bQBnTJW2BOc

Dotstar LED(s)

In Duet firmware, a driver for DotStar LED strips is available in the latest RRF source code for Duet3 only, not for Duet2…. These LED strips use standard SPI MOSI and SCLK signals, so not only does the processor not need to give them its undivided attention, it can use DMA to send the data. To use them on the Duet WiFi or Duet Ethernet, you would need to make a hardware adapter to level shift the MOSI and SCLK signals to 5V, also to gate SCLK with your chosen /CS pin. You could do all of this using a single 74HCT02 chip. You would also need to enable the DotStar support for the Duet 2 build of RRF3.X, via a new firmware build (with Platformio).