Configuration.h

MARLIN 2.0 – Software for 3D printer

Januar 2022
Nu er det på tide at forklare lidt om den software, jeg benytter til at styre 3D printerne med. Software er temmelig omfangsrig, og kan DOWNLOADES på denne hjemmeside: http://marlinfw.org/

For lige at gøre det klart, så indeholder softwaren en standard opsætning, og det eneste sted du som bruger af softwaren, skal definere din 3D printers minimumsdefinitioner er i de 2 filer:

1: Configuration.h
2: Configuration_adv.h

Heraf er nr. 1 den vigtigste, for det er her der vælges de forudsætninger, der skal til for, at din 3D printer kan printe. 3D printere kan defineres med en lang række af forskellig hardware, og i mine 3D printere har jeg konsekvent brugt **Arduino ATMEGA 2560 motherboard**, og **RAMPS 1.4** som interface til printerens øvrige hardware. Desuden er denne **LILLE 3D printer** bestykket med **POLOLU 4988 Stepper drivere.** Display hardware har jeg ligeledes konsekvent valgt REPRAP.me's såkaldte **RepRapDiscount FULL GRAPHIC Smart Controller** et levn fra min begyndelse af interessen for 3D printer, alt indkøbt i Kina via www.ebay.com.

I **Configuration_adv.h** er det meget begrænset, hvad der skal foretages af ændringer. Dog er det nødvendigt at lave en lille tilføjelse i forbindelse med aktivering af **"Filament Runout".** Hvis du ikke får lavet tilføjelsen i første kompilering, vil ARDUINO IDE specifikt bede om at få tilføjet rettelse og henvise til, hvor rettelsen skal foretages, og derfor vil jeg ikke her komme nærmere ind på dette.

Kompilering af MARLIN kan ske med ARDUINO IDE, men er i MARLIN ver. 2 temmelig lang tid om at arbejde sig igennem proceduren. Den skal nok blive færdig, selv om det kan tage 5-8 minutter eller mere, lidt afhængig af din computers kapacitet.
Tålmodighed er her en dyd.

I min gennemgang af **Configuration.h** herunder, har jeg farvelagt de ændringer eller tilføjelser, som jeg har berørt med RØD, og som er nødvendige for at få min printer til at køre.

For at aktivere en funktion, er det nødvendigt at fjerne "//" foran "#define…".
Configuration.h er forud defineret med en række funktioner som er nødvendige for at få en 3D printer til at 3D printe, men kun de funktioner med RØD har jeg lavet ændringer ved eller aktiveret. **FED** tekst eller tal markerer få steder i min valgte konfiguration.

I lighed med LINKET til MARLIN's hjemmeside øverst i denne tekst, vil der flere steder i **Configuration.h** være LINKS til hjemmesider, hvor et enkelt emne i **Configuration.h** blive beskrevet nærmere, eller hvor en bestemt hardware del kan indkøbes. Disse LINKS vil være markeret med *BLÅ farve*.

At denne gennemgang af **Configuration.h** fylder 46 sider skyldes primært, at den meget langt hen ad vejen, er selvforklarende. Der er for langt de fleste funktioner angivet, hvad der skal til for, at du som bruger af softwaren er i stand til at finde den rigtige definition af de enkelte funktioner.

Configuration.h

Jeg har valgt altid at lægge resultatet af compilering ind øverst i **Configuration.h**. Hvis jeg senere vil foretage ændringer, har jeg stadig overblik over hukommelsesforbruget, og anden information, som her, hvor jeg nævner, at denne version er beregnet til **"Den LILLE 3D printer".**

Jeg håber, at du ved gennemgang af denne **Configuration.h**, får erfaring nok til, at din hjemmebyggede 3D printer kommer op og køre. Hvis ikke, så må du gerne kontakte mig:
e-mail: oz6ym@planker.dk

```
/** OZ6YM...: CONFIGURATION_H_VERSION 02000902 – ARDUINO IDE Compiling version - Just wait, it takes time...
 *  For LILLE HOMEMADE 3D-printer - Efterår og vinter 2021
 *  ========================================================================
 *  Sketch uses 136260 bytes (53%) of program storage space. Maximum is 253952 bytes.
 *  Global variables use 4996 bytes (60%) of dynamic memory, leaving 3196 ,bytes for local variables.
 *  Maximum is 8192 bytes.
 *  ========================================================================

 * Marlin 3D Printer Firmware
 * Copyright (c) 2020 MarlinFirmware [ https://github.com/MarlinFirmware/Marlin ]
 *
 * Based on Sprinter and grbl.
 * Copyright (c) 2011 Camiel Gubbels / Erik van der Zalm
 *
 * This program is free software: you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation, either version 3 of the License, or
 * (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program.  If not, see < https://www.gnu.org/licenses >.
 *
 */
#pragma once
/**
 * Configuration.h
 *
 * Basic settings such as:
 *
 * - Type of electronics
 * - Type of temperature sensor
 * - Printer geometry
 * - Endstop configuration
 * - LCD controller
 * - Extra features
 *
 * Advanced settings can be found in Configuration_adv.h
 */
#define CONFIGURATION_H_VERSION 02000902
//===========================================================
//============================ Getting Started  =============================
//===========================================================
/**
 * Here are some useful links to help get your machine configured and calibrated:
 *
 * Example Configs:     https://github.com/MarlinFirmware/Configurations/branches/all
 *
```

```
 * Průša Calculator:     https://blog.prusaprinters.org/calculator_3416/
 *
 * Calibration Guides:  https://reprap.org/wiki/Calibration
 *                      https://reprap.org/wiki/Triffid_Hunter%27s_Calibration_Guide
 *                      https://sites.google.com/site/repraplogphase/calibration-of-your-reprap
 *                      https://youtu.be/wAL9d7FgInk
 *
 * Calibration Objects: https://www.thingiverse.com/thing:5573
 *                      https://www.thingiverse.com/thing:1278865
 */
//===========================================================
//==================== DELTA / SCARA / TPARA ================
//===========================================================
//
// Download configurations from the link above and customize for your machine.
// Examples are located in config/examples/delta, .../SCARA, and .../TPARA.
//
//===========================================================

// @section info
// Author info of this build printed to the host during boot and M115
#define STRING_CONFIG_H_AUTHOR "OZ6YM, Palle Andersen" // Who made the changes.
//#define CUSTOM_VERSION_FILE Version.h // Path from the root directory (no quotes)
/**
 * *** VENDORS PLEASE READ ***
 *
 * Marlin allows you to add a custom boot image for Graphical LCDs.
 * With this option Marlin will first show your custom screen followed
 * by the standard Marlin logo with version number and web URL.
 *
 * We encourage you to take advantage of this new feature and we also
 * respectfully request that you retain the unmodified Marlin boot screen.
 */

// Show the Marlin bootscreen on startup. ** ENABLE FOR PRODUCTION **
#define SHOW_BOOTSCREEN
// Show the bitmap in Marlin/_Bootscreen.h on startup.
//#define SHOW_CUSTOM_BOOTSCREEN
// Show the bitmap in Marlin/_Statusscreen.h on the status screen.
//#define CUSTOM_STATUS_SCREEN_IMAGE
// @section machine
/**
 * Select the serial port on the board to use for communication with the host.
 * This allows the connection of wireless adapters (for instance) to non-default port pins.
 * Serial port -1 is the USB emulated serial port, if available.
 * Note: The first serial port (-1 or 0) will always be used by the Arduino bootloader.
 *
 * :[-1, 0, 1, 2, 3, 4, 5, 6, 7]
 */
#define SERIAL_PORT 0
/**
 * Serial Port Baud Rate
 * This is the default communication speed for all serial ports.
 * Set the baud rate defaults for additional serial ports below.
 *
 * 250000 works in most cases, but you might try a lower speed if
 * you commonly experience drop-outs during host printing.
 * You may try up to 1000000 to speed up SD file transfer.
 *
 * :[2400, 9600, 19200, 38400, 57600, 115200, 250000, 500000, 1000000]
 */
```

```
#define BAUDRATE 115200
//#define BAUD_RATE_GCODE     // Enable G-code M575 to set the baud rate
/**
 * Select a secondary serial port on the board to use for communication with the host.
 * Currently Ethernet (-2) is only supported on Teensy 4.1 boards.
 * :[-2, -1, 0, 1, 2, 3, 4, 5, 6, 7]

 */
//#define SERIAL_PORT_2 -1
//#define BAUDRATE_2 250000   // Enable to override BAUDRATE
/**
 * Select a third serial port on the board to use for communication with the host.
 * Currently only supported for AVR, DUE, LPC1768/9 and STM32/STM32F1
 * :[-1, 0, 1, 2, 3, 4, 5, 6, 7]
 */
//#define SERIAL_PORT_3 1
//#define BAUDRATE_3 250000   // Enable to override BAUDRATE
// Enable the Bluetooth serial interface on AT90USB devices
//#define BLUETOOTH
// Choose the name from boards.h that matches your setup
#ifndef MOTHERBOARD
  #define MOTHERBOARD BOARD_RAMPS_14_EFF   // Ramps 1.4 med FAN 1 og FAN 2 aktive
#endif
// Name displayed in the LCD "Ready" message and Info menu
#define CUSTOM_MACHINE_NAME "LILLE HM 3D"
// Printer's unique ID, used by some programs to differentiate between machines.
// Choose your own or use a service like https://www.uuidgenerator.net/version4
//#define MACHINE_UUID "00000000-0000-0000-0000-000000000000"
/**
 * Define the number of coordinated linear axes.
 * See https://github.com/DerAndere1/Marlin/wiki
 * Each linear axis gets its own stepper control and endstop:
 *
 *   Steppers: *_STEP_PIN, *_ENABLE_PIN, *_DIR_PIN, *_ENABLE_ON
 *   Endstops: *_STOP_PIN, USE_*MIN_PLUG, USE_*MAX_PLUG
 *       Axes: *_MIN_POS, *_MAX_POS, INVERT_*_DIR
 *    Planner: DEFAULT_AXIS_STEPS_PER_UNIT, DEFAULT_MAX_FEEDRATE
 *             DEFAULT_MAX_ACCELERATION, AXIS_RELATIVE_MODES,
 *             MICROSTEP_MODES, MANUAL_FEEDRATE
 *
 * :[3, 4, 5, 6]
 */
//#define LINEAR_AXES 3
/**
 * Axis codes for additional axes:
 * This defines the axis code that is used in G-code commands to
 * reference a specific axis.
 * 'A' for rotational axis parallel to X
 * 'B' for rotational axis parallel to Y
 * 'C' for rotational axis parallel to Z
 * 'U' for secondary linear axis parallel to X
 * 'V' for secondary linear axis parallel to Y
 * 'W' for secondary linear axis parallel to Z
 * Regardless of the settings, firmware-internal axis IDs are
 * I (AXIS4), J (AXIS5), K (AXIS6).
 */
#if LINEAR_AXES >= 4
  #define AXIS4_NAME 'A' // :['A', 'B', 'C', 'U', 'V', 'W']
#endif
#if LINEAR_AXES >= 5
  #define AXIS5_NAME 'B' // :['A', 'B', 'C', 'U', 'V', 'W']
```

```
#endif
#if LINEAR_AXES >= 6
  #define AXIS6_NAME 'C' // :['A', 'B', 'C', 'U', 'V', 'W']
#endif
// @section extruder
// This defines the number of extruders
// :[0, 1, 2, 3, 4, 5, 6, 7, 8]
#define EXTRUDERS 1
// Generally expected filament diameter (1.75, 2.85, 3.0, ...). Used for Volumetric, Filament // // Width Sensor, etc.
#define DEFAULT_NOMINAL_FILAMENT_DIA 1.75
// For Cyclops or any "multi-extruder" that shares a single nozzle.
//#define SINGLENOZZLE
// Save and restore temperature and fan speed on tool-change.
// Set standby for the unselected tool with M104/106/109 T...
#if ENABLED(SINGLENOZZLE)
  //#define SINGLENOZZLE_STANDBY_TEMP
  //#define SINGLENOZZLE_STANDBY_FAN
#endif
/**
 * Multi-Material Unit
 * Set to one of these predefined models:
 *
 *   PRUSA_MMU1          : Průša MMU1 (The "multiplexer" version)
 *   PRUSA_MMU2          : Průša MMU2
 *   PRUSA_MMU2S         : Průša MMU2S (Requires MK3S extruder with motion sensor,
 *                          XTRUDERS = 5)
 *   EXTENDABLE_EMU_MMU2   : MMU with configurable number of filaments (ERCF, SMuFF or
 *                           similar with Průša MMU2 compatible firmware)
 *   EXTENDABLE_EMU_MMU2S : MMUS with configurable number of filaments (ERCF, SMuFF
 *                           or similar with Průša MMU2 compatible firmware)
 *
 * Requires NOZZLE_PARK_FEATURE to park print head in case MMU unit fails.
 * See additional options in Configuration_adv.h.
 */
//#define MMU_MODEL PRUSA_MMU2
// A dual extruder that uses a single stepper motor
//#define SWITCHING_EXTRUDER
#if ENABLED(SWITCHING_EXTRUDER)
  #define SWITCHING_EXTRUDER_SERVO_NR 0
  #define SWITCHING_EXTRUDER_SERVO_ANGLES { 0, 90 } // Angles for E0, E1[, E2, E3]
  #if EXTRUDERS > 3
    #define SWITCHING_EXTRUDER_E23_SERVO_NR 1
  #endif
#endif
// A dual-nozzle that uses a servomotor to raise/lower one (or both) of the nozzles
//#define SWITCHING_NOZZLE
#if ENABLED(SWITCHING_NOZZLE)
  #define SWITCHING_NOZZLE_SERVO_NR 0
  //#define SWITCHING_NOZZLE_E1_SERVO_NR 1        // If two servos are used, the index
                                                  // of the second
  #define SWITCHING_NOZZLE_SERVO_ANGLES { 0, 90 }   // Angles for E0, E1 (single servo)
                                                    // or lowered/raised (dual servo)
#endif
/**
 * Two separate X-carriages with extruders that connect to a moving part
 * via a solenoid docking mechanism. Requires SOL1_PIN and SOL2_PIN.
 */
//#define PARKING_EXTRUDER
/**
 * Two separate X-carriages with extruders that connect to a moving part
 * via a magnetic docking mechanism using movements and no solenoid
```

```
 *
 * project       :    https://www.thingiverse.com/thing:3080893
 * movements :    https://youtu.be/0xCEiG9VS3k
 *                    https://youtu.be/Bqbcs0CU2FE
 */
//#define MAGNETIC_PARKING_EXTRUDER
#if EITHER(PARKING_EXTRUDER, MAGNETIC_PARKING_EXTRUDER)
  #define PARKING_EXTRUDER_PARKING_X { -78, 184 }     // X positions for parking the extruders
  #define PARKING_EXTRUDER_GRAB_DISTANCE 1           // (mm) Distance to move beyond the parking point to
                                                     // grab the extruder
  //#define MANUAL_SOLENOID_CONTROL     // Manual control of docking solenoids with M380 S / M381
  #if ENABLED(PARKING_EXTRUDER)
    #define PARKING_EXTRUDER_SOLENOIDS_INVERT       // If enabled, the solenoid is NOT
                                                    //magnetized with applied voltage
    #define PARKING_EXTRUDER_SOLENOIDS_PINS_ACTIVE LOW  // LOW or HIGH pin signal energizes the coil
    #define PARKING_EXTRUDER_SOLENOIDS_DELAY 250  // (ms) Delay for magnetic field.
                                                  // No delay if 0 or not defined.
    //#define MANUAL_SOLENOID_CONTROL               // Manual control of docking solenoids with M380 S / M381
  #elif ENABLED(MAGNETIC_PARKING_EXTRUDER)
    #define MPE_FAST_SPEED 9000      // (mm/min) Speed for travel before last distance point
    #define MPE_SLOW_SPEED 4500      // (mm/min) Speed for last distance travel to park and couple
    #define MPE_TRAVEL_DISTANCE 10 // (mm) Last distance point
    #define MPE_COMPENSATION 0 // Offset Compensation -1 , 0 , 1 (multiplier) only for coupling
  #endif
#endif
/**
 * Switching Toolhead
 *
 * Support for swappable and dockable toolheads, such as
 * the E3D Tool Changer. Toolheads are locked with a servo.
 */
//#define SWITCHING_TOOLHEAD
/**
 * Magnetic Switching Toolhead
 *
 * Support swappable and dockable toolheads with a magnetic
 * docking mechanism using movement and no servo.
 */
//#define MAGNETIC_SWITCHING_TOOLHEAD
/**
 * Electromagnetic Switching Toolhead
 *
 * Parking for CoreXY / HBot kinematics.
 * Toolheads are parked at one edge and held with an electromagnet.
 * Supports more than 2 Toolheads. See https://youtu.be/JolbsAKTKf4
 */
//#define ELECTROMAGNETIC_SWITCHING_TOOLHEAD
#if ANY(SWITCHING_TOOLHEAD, MAGNETIC_SWITCHING_TOOLHEAD, ELECTROMAGNETIC_SWITCHING_TOOLHEAD)
  #define SWITCHING_TOOLHEAD_Y_POS 235         // (mm) Y position of the toolhead dock
  #define SWITCHING_TOOLHEAD_Y_SECURITY 10  // (mm) Security distance Y axis
  #define SWITCHING_TOOLHEAD_Y_CLEAR  60      // (mm) Minimum distance from dock for nobstructed X axis
  #define SWITCHING_TOOLHEAD_X_POS         { 215, 0 }  // (mm) X positions for parking the extruders
  #if ENABLED(SWITCHING_TOOLHEAD)
    #define SWITCHING_TOOLHEAD_SERVO_NR       2        // Index of the servo connector
    #define SWITCHING_TOOLHEAD_SERVO_ANGLES { 0, 180 }  // (degrees) Angles for Lock, Unlock
    #elif ENABLED(MAGNETIC_SWITCHING_TOOLHEAD)
    #define SWITCHING_TOOLHEAD_Y_RELEASE      5        // (mm) Security distance Y axis
    #define SWITCHING_TOOLHEAD_X_SECURITY   { 90, 150 } // (mm) Security distance X axis (T0,T1)
    //#define PRIME_BEFORE_REMOVE          // Prime the nozzle before release from the dock
    #if ENABLED(PRIME_BEFORE_REMOVE)
      #define SWITCHING_TOOLHEAD_PRIME_MM     20  // (mm)   Extruder prime length
```

```
      #define SWITCHING_TOOLHEAD_RETRACT_MM 10  // (mm)   Retract after priming length
      #define SWITCHING_TOOLHEAD_PRIME_FEEDRATE 300 // (mm/min) Extruder prime federate
    #define SWITCHING_TOOLHEAD_RETRACT_FEEDRATE 2400  // (mm/min) Extruder retract federate
    #endif
  #elif ENABLED(ELECTROMAGNETIC_SWITCHING_TOOLHEAD)
    #define SWITCHING_TOOLHEAD_Z_HOP          2         // (mm) Z raise for switching
  #endif
#endif
/**
 * "Mixing Extruder"
 *   - Adds G-codes M163 and M164 to set and "commit" the current mix factors.
 *   - Extends the stepping routines to move multiple steppers in proportion to the mix.
 *   - Optional support for Repetier Firmware's 'M164 S<index>' supporting virtual tools.
 *   - This implementation supports up to two mixing extruders.
 *   - Enable DIRECT_MIXING_IN_G1 for M165 and mixing in G1 (from Pia Taubert's reference
 *     implementation).
 */
//#define MIXING_EXTRUDER
#if ENABLED(MIXING_EXTRUDER)
  #define MIXING_STEPPERS 2        // Number of steppers in your mixing extruder
  #define MIXING_VIRTUAL_TOOLS 16  // Use the Virtual Tool method with M163 and M164
  //#define DIRECT_MIXING_IN_G1    // Allow ABCDHI mix factors in G1 movement commands
  //#define GRADIENT_MIX           // Support for gradient mixing with M166 and LCD
  //#define MIXING_PRESETS         // Assign 8 default V-tool presets for 2 or 3 MIXING_STEPPERS
  #if ENABLED(GRADIENT_MIX)
    //#define GRADIENT_VTOOL       // Add M166 T to use a V-tool index as a Gradient alias
  #endif
#endif
// Offset of the extruders (uncomment if using more than one and relying on firmware to
// position when changing).
// The offset has to be X=0, Y=0 for the extruder 0 hotend (default extruder).
// For the other hotends it is their distance from the extruder 0 hotend.
//#define HOTEND_OFFSET_X { 0.0, 20.00 } // (mm) relative X-offset for each nozzle
//#define HOTEND_OFFSET_Y { 0.0, 5.00 }  // (mm) relative Y-offset for each nozzle
//#define HOTEND_OFFSET_Z { 0.0, 0.00 }  // (mm) relative Z-offset for each nozzle
// @section machine
/**
 * Power Supply Control
 *
 * Enable and connect the power supply to the PS_ON_PIN.
 * Specify whether the power supply is active HIGH or active LOW.
 */
//#define PSU_CONTROL
//#define PSU_NAME "Power Supply"
#if ENABLED(PSU_CONTROL)
  //#define MKS_PWC               // Using the MKS PWC add-on
  //#define PS_OFF_CONFIRM          // Confirm dialog when power off
  //#define PS_OFF_SOUND            // Beep 1s when power off
  #define PSU_ACTIVE_STATE LOW      // Set 'LOW' for ATX, 'HIGH' for X-Box
  //#define PSU_DEFAULT_OFF         // Keep power off until enabled directly with M80
  //#define PSU_POWERUP_DELAY 250   // (ms) Delay for the PSU to warm up to full power
  //#define PSU_POWERUP_GCODE  "M355 S1"  // G-code to run after power-on (e.g., case light on)
  //#define PSU_POWEROFF_GCODE "M355 S0"  // G-code to run before power-off (e.g., case light off)
  //#define AUTO_POWER_CONTROL      // Enable automatic control of the PS_ON pin
  #if ENABLED(AUTO_POWER_CONTROL)
    #define AUTO_POWER_FANS         // Turn on PSU if fans need power
    #define AUTO_POWER_E_FANS
    #define AUTO_POWER_CONTROLLERFAN
    #define AUTO_POWER_CHAMBER_FAN
    #define AUTO_POWER_COOLER_FAN
    //#define AUTO_POWER_E_TEMP       50 // (°C) Turn on PSU if any extruder is over this temperature
```

```
    //#define AUTO_POWER_CHAMBER_TEMP  30 // (°C) Turn on PSU if the chamber is over this temperature
    //#define AUTO_POWER_COOLER_TEMP   26 // (°C) Turn on PSU if the cooler is over this temperature
    #define POWER_TIMEOUT 30 // (s) Turn off power if the machine is idle for this duration
    //#define POWER_OFF_DELAY         60 // (s) Delay of poweroff after M81 command.
                                         // Useful to let fans run for extra time.
  #endif
#endif
//===========================================================
//===================== Thermal Settings ====================
//===========================================================
// @section temperature
//**
/* --NORMAL IS 4.7kΩ PULLUP!-- 1kΩ pullup can be used on hotend sensor, using correct
 * resistor and table
 *
 * Temperature sensors available:
 *
 *  SPI RTD/Thermocouple Boards - IMPORTANT: Read the NOTE below!
 *  -------
 *    -5 : MAX31865 with Pt100/Pt1000, 2, 3, or 4-wire  (only for sensors 0-1)
 *                NOTE: You must uncomment/set the MAX31865_*_OHMS_n defines below.
 *    -3 : MAX31855 with Thermocouple, -200°C to +700°C (only for sensors 0-1)
 *    -2 : MAX6675  with Thermocouple, 0°C to +700°C    (only for sensors 0-1)
 *
 *  NOTE: Ensure TEMP_n_CS_PIN is set in your pins file for each TEMP_SENSOR_n using an
 *  SPI Thermocouple. By default,
 *       Hardware SPI on the default serial bus is used. If you have also set TEMP_n_SCK_PIN
 *  and TEMP_n_MISO_PIN, Software SPI will be used on those ports instead. You can force
 *  Hardware SPI on the default bus in the Configuration_adv.h file.
 *  At this time, separate Hardware SPI buses for sensors are not supported.
 *
 *  Analog Themocouple Boards
 *  -------
 *    -4 : AD8495 with Thermocouple
 *    -1 : AD595  with Thermocouple
 *
 *  Analog Thermistors - 4.7kΩ pullup – Normal
 *  -------
 *    1 : 100kΩ  EPCOS - Best choice for EPCOS thermistors
 *  331 : 100kΩ  Same as #1, but 3.3V scaled for MEGA
 *  332 : 100kΩ  Same as #1, but 3.3V scaled for DUE
 *    2 : 200kΩ  ATC Semitec 204GT-2
 *  202 : 200kΩ  Copymaster 3D
 *    3 : ???Ω   Mendel-parts thermistor
 *    4 : 10kΩ   Generic Thermistor !! DO NOT use for a hotend - it gives bad resolution at
 *                high temp. !!
 *    5 : 100kΩ  ATC Semitec 104GT-2/104NT-4-R025H42G - Used in ParCan, J-Head, and
 *          E3D, SliceEngineering 300°C
 *  501 : 100kΩ  Zonestar - Tronxy X3A
 *  502 : 100kΩ  Zonestar - used by hot bed in Zonestar Průša P802M
 *  512 : 100kΩ  RPW-Ultra hotend
 *    6 : 100kΩ  EPCOS - Not as accurate as table #1 (created using a fluke thermocouple)
 *    7 : 100kΩ  Honeywell 135-104LAG-J01
 *   71 : 100kΩ  Honeywell 135-104LAF-J01
 *    8 : 100kΩ  Vishay 0603 SMD NTCS0603E3104FXT
 *    9 : 100kΩ  GE Sensing AL03006-58.2K-97-G1
 *   10 : 100kΩ  RS PRO 198-961
 *   11 : 100kΩ  Keenovo AC silicone mats, most Wanhao i3 machines - beta 3950, 1%
 *   12 : 100kΩ  Vishay 0603 SMD NTCS0603E3104FXT (#8) - calibrated for Makibox hot bed
 *   13 : 100kΩ  Hisens up to 300°C - for "Simple ONE" & "All In ONE" hotend - beta 3950,1%
 *   15 : 100kΩ  Calibrated for JGAurora A5 hotend
```

```
*   18 : 200kΩ  ATC Semitec 204GT-2 Dagoma.Fr - MKS_Base_DKU001327
*   22 : 100kΩ  GTM32 Pro vB - hotend - 4.7kΩ pullup to 3.3V and 220Ω to analog input
*   23 : 100kΩ  GTM32 Pro vB - bed - 4.7kΩ pullup to 3.3v and 220Ω to analog input
*   30 : 100kΩ  Kis3d Silicone heating mat 200W/300W with 6mm precision cast plate (EN
*                AW 5083) NTC100K - beta 3950
*   60 : 100kΩ  Maker's Tool Works Kapton Bed Thermistor - beta 3950
*   61 : 100kΩ  Formbot/Vivedino 350°C Thermistor - beta 3950
*   66 : 4.7MΩ  Dyze Design High Temperature Thermistor
*   67 : 500kΩ  SliceEngineering 450°C Thermistor
*   70 : 100kΩ  bq Hephestos 2
*   75 : 100kΩ  Generic Silicon Heat Pad with NTC100K MGB18-104F39050L32
* 2000 : 100kΩ  Ultimachine Rambo TDK NTCG104LH104KT1 NTC100K motherboard
*                Thermistor
*
* Analog Thermistors - 1kΩ pullup - Atypical, and requires changing out the 4.7kΩ pullup for 1kΩ.
* -------                    (but gives greater accuracy and more stable PID)
*   51 : 100kΩ  EPCOS (1kΩ pullup)
*   52 : 200kΩ  ATC Semitec 204GT-2 (1kΩ pullup)
*   55 : 100kΩ  ATC Semitec 104GT-2 - Used in ParCan & J-Head (1kΩ pullup)
*
* Analog Thermistors - 10kΩ pullup – Atypical
* -------
*   99 : 100kΩ  Found on some Wanhao i3 machines with a 10kΩ pull-up resistor
*
* Analog RTDs (Pt100/Pt1000)
* -------
*  110 : Pt100  with 1kΩ pullup (atypical)
*  147 : Pt100  with 4.7kΩ pullup
* 1010 : Pt1000 with 1kΩ pullup (atypical)
* 1047 : Pt1000 with 4.7kΩ pullup (E3D)
*   20 : Pt100  with circuit in the Ultimainboard V2.x with mainboard ADC reference voltage =
*                INA826 amplifier-board supply voltage.
*            NOTE: (1) Must use an ADC input with no pullup.
*                  (2) Some INA826 amplifiers are unreliable at 3.3V so consider using
*                      sensor 147, 110, or 21.
*   21 : Pt100  with circuit in the Ultimainboard V2.x with 3.3v ADC reference voltage
*        (STM32, LPC176x....) and 5V INA826 amplifier board supply.
*            NOTE: ADC pins are not 5V tolerant. Not recommended because it's possible to
*                  damage the CPU by going over 500°C.
*  201 : Pt100  with circuit in Overlord, similar to Ultimainboard V2.x
*
* Custom/Dummy/Other Thermal Sensors
* ------
*    0 : not used
* 1000 : Custom - Specify parameters in Configuration_adv.h
*
*  !!! Use these for Testing or Development purposes. NEVER for production machine. !!!
*  998 : Dummy Table that ALWAYS reads 25°C or the temperature defined below.
*  999 : Dummy Table that ALWAYS reads 100°C or the temperature defined below.
*
*/
#define TEMP_SENSOR_0 1
#define TEMP_SENSOR_1 0
#define TEMP_SENSOR_2 0
#define TEMP_SENSOR_3 0
#define TEMP_SENSOR_4 0
#define TEMP_SENSOR_5 0
#define TEMP_SENSOR_6 0
#define TEMP_SENSOR_7 0
#define TEMP_SENSOR_BED 0    // NO BED HEADER
#define TEMP_SENSOR_PROBE 0
```

```
#define TEMP_SENSOR_CHAMBER 0
#define TEMP_SENSOR_COOLER 0
#define TEMP_SENSOR_BOARD 0
#define TEMP_SENSOR_REDUNDANT 0
// Dummy thermistor constant temperature readings, for use with 998 and 999
#define DUMMY_THERMISTOR_998_VALUE  25
#define DUMMY_THERMISTOR_999_VALUE 100
// Resistor values when using MAX31865 sensors (-5) on TEMP_SENSOR_0 / 1
//#define MAX31865_SENSOR_OHMS_0      100   // (Ω) Typically 100 or 1000 (PT100 or PT1000)

//#define MAX31865_CALIBRATION_OHMS_0 430   // (Ω) Typically 430 for Adafruit PT100;
                                            // 4300 for Adafruit PT1000
//#define MAX31865_SENSOR_OHMS_1      100
//#define MAX31865_CALIBRATION_OHMS_1 430
#define TEMP_RESIDENCY_TIME 10  // (seconds) Time to wait for hotend to "settle" in M109
#define TEMP_WINDOW 1  // (°C) Temperature proximity for the "temperature reached" timer
#define TEMP_HYSTERESIS 3 // (°C) Temperature proximity considered "close nough" to the target
#define TEMP_BED_RESIDENCY_TIME 10  // (seconds) Time to wait for bed to "settle" in M190
#define TEMP_BED_WINDOW             1  // (°C) Temperature proximity for the "temperature reached" timer
#define TEMP_BED_HYSTERESIS         3  // (°C) Temperature proximity considered "close enough" to the target
#define TEMP_CHAMBER_RESIDENCY_TIME 10  // (seconds) Time to wait for chamber to "settle" in M191
#define TEMP_CHAMBER_WINDOW         1  // (°C) Temperature proximity for the "temperature reached" timer
#define TEMP_CHAMBER_HYSTERESIS      3  // (°C) Temperature proximity considered "close enough" to the target
/**
 * Redundant Temperature Sensor (TEMP_SENSOR_REDUNDANT)
 *
 * Use a temp sensor as a redundant sensor for another reading. Select an unused temperature sensor, and another
 * sensor you'd like it to be redundant for. If the two thermistors differ by TEMP_SENSOR_REDUNDANT_MAX_DIFF
 * °C), the print will be aborted. Whichever sensor is selected will have its normal functions disabled; i.e. selecting
 * the Bed sensor (-1) will disable bed heating/monitoring.
 *
 * For selecting source/target use: COOLER, PROBE, BOARD, CHAMBER, BED, E0, E1, E2, E3, E4, E5, E6, E7
 */
#if TEMP_SENSOR_REDUNDANT
  #define TEMP_SENSOR_REDUNDANT_SOURCE    E1  // The sensor that will provide the redundant reading.
  #define TEMP_SENSOR_REDUNDANT_TARGET    E0  // The sensor that we are providing a redundant reading for.
  #define TEMP_SENSOR_REDUNDANT_MAX_DIFF  10  // (°C) Temperature difference that will trigger a print abort.
#endif
// Below this temperature the heater will be switched off
// because it probably indicates a broken thermistor wire.
#define HEATER_0_MINTEMP   5
#define HEATER_1_MINTEMP   5
#define HEATER_2_MINTEMP   5
#define HEATER_3_MINTEMP   5
#define HEATER_4_MINTEMP   5
#define HEATER_5_MINTEMP   5
#define HEATER_6_MINTEMP   5
#define HEATER_7_MINTEMP   5
#define BED_MINTEMP       5
#define CHAMBER_MINTEMP    5
// Above this temperature the heater will be switched off.
// This can protect components from overheating, but NOT from shorts and failures.
// (Use MINTEMP for thermistor short/failure protection.)
#define HEATER_0_MAXTEMP 275
#define HEATER_1_MAXTEMP 275
#define HEATER_2_MAXTEMP 275
#define HEATER_3_MAXTEMP 275
#define HEATER_4_MAXTEMP 275
#define HEATER_5_MAXTEMP 275
#define HEATER_6_MAXTEMP 275
#define HEATER_7_MAXTEMP 275
```

```
#define BED_MAXTEMP      150
#define CHAMBER_MAXTEMP  60
/**
 * Thermal Overshoot
 * During heatup (and printing) the temperature can often "overshoot" the target by many degrees
 * (especially before PID tuning). Setting the target temperature too close to MAXTEMP guarantees
 * a MAXTEMP shutdown! Use these values to forbid temperatures being set too close to MAXTEMP.
 */
#define HOTEND_OVERSHOOT 15   // (°C) Forbid temperatures over MAXTEMP – OVERSHOOT
#define BED_OVERSHOOT    10   // (°C) Forbid temperatures over MAXTEMP – OVERSHOOT
#define COOLER_OVERSHOOT  2   // (°C) Forbid temperatures closer than OVERSHOOT
//===========================================================================
//============================= PID Settings ================================
//===========================================================================
// PID Tuning Guide here: https://reprap.org/wiki/PID_Tuning
// Comment the following line to disable PID and enable bang-bang.
#define PIDTEMP
#define BANG_MAX 255     // Limits current to nozzle while in bang-bang mode; 255=full current
#define PID_MAX BANG_MAX // Limits current to nozzle while PID is active (see PID_FUNCTIONAL_RANGE below);
                         //  255=full current
#define PID_K1 0.95      // Smoothing factor within any PID loop
#if ENABLED(PIDTEMP)
  //#define PID_EDIT_MENU        // Add PID editing to the "Advanced Settings" menu. (~700 bytes of PROGMEM)
  //#define PID_AUTOTUNE_MENU     // Add PID auto-tuning to the "Advanced Settings" menu. (~250 bytes of
                                  // PROGMEM)
  //#define PID_PARAMS_PER_HOTEND // Uses separate PID parameters for each extruder (useful for mismatched
                                  //extruders)
                                  // Set/get with gcode: M301 E[extruder number, 0-2]
  #if ENABLED(PID_PARAMS_PER_HOTEND)
    // Specify up to one value per hotend here, according to your setup.
    // If there are fewer values, the last one applies to the remaining hotends.
    #define DEFAULT_Kp_LIST {  22.20,  22.20 }
    #define DEFAULT_Ki_LIST {   1.08,   1.08 }
    #define DEFAULT_Kd_LIST { 114.00, 114.00 }
  #else
    #define DEFAULT_Kp  22.20
    #define DEFAULT_Ki   1.08
    #define DEFAULT_Kd 114.00
  #endif
#endif // PIDTEMP
//===========================================================================
//====================== PID > Bed Temperature Control ======================
//===========================================================================
/**
 * PID Bed Heating
 *
 * If this option is enabled set PID constants below.
 * If this option is disabled, bang-bang will be used and BED_LIMIT_SWITCHING will enable hysteresis.
 *
 * The PID frequency will be the same as the extruder PWM.
 * If PID_dT is the default, and correct for the hardware/configuration, that means 7.689Hz,
 * which is fine for driving a square wave into a resistive load and does not significantly
 * impact FET heating. This also works fine on a Fotek SSR-10DA Solid State Relay into a 250W
 * heater. If your configuration is significantly different than this and you don't understand
 * the issues involved, don't use bed PID until someone else verifies that your hardware works.
 */
//#define PIDTEMPBED
//#define BED_LIMIT_SWITCHING
/**
 * Max Bed Power
 * Applies to all forms of bed control (PID, bang-bang, and bang-bang with hysteresis).
```

```
 * When set to any value below 255, enables a form of PWM to the bed that acts like a divider
 * so don't use it unless you are OK with PWM on your bed. (See the comment on enabling PIDTEMPBED)
 */
#define MAX_BED_POWER 255 // limits duty cycle to bed; 255=full current
#if ENABLED(PIDTEMPBED)
  //#define MIN_BED_POWER 0
  //#define PID_BED_DEBUG // Sends debug data to the serial port.
  // 120V 250W silicone heater into 4mm borosilicate (MendelMax 1.5+)
  // from FOPDT model - kp=.39 Tp=405 Tdead=66, Tc set to 79.2, aggressive factor of .15 (vs .1, 1, 10)
  #define DEFAULT_bedKp 10.00
  #define DEFAULT_bedKi .023
  #define DEFAULT_bedKd 305.4
  // FIND YOUR OWN: "M303 E-1 C8 S90" to run autotune on the bed at 90 degreesC for 8 cycles.
#endif // PIDTEMPBED
//===========================================================================
//========================== PID > Chamber Temperature Control ==============
//===========================================================================
/**
 * PID Chamber Heating
 *
 * If this option is enabled set PID constants below.
 * If this option is disabled, bang-bang will be used and CHAMBER_LIMIT_SWITCHING will enable
 * hysteresis.
 *
 * The PID frequency will be the same as the extruder PWM.
 * If PID_dT is the default, and correct for the hardware/configuration, that means 7.689Hz,
 * which is fine for driving a square wave into a resistive load and does not significantly
 * impact FET heating. This also works fine on a Fotek SSR-10DA Solid State Relay into a 200W
 * heater. If your configuration is significantly different than this and you don't understand
 * the issues involved, don't use chamber PID until someone else verifies that your hardware works.
 */
//#define PIDTEMPCHAMBER
//#define CHAMBER_LIMIT_SWITCHING
/**
 * Max Chamber Power
 * Applies to all forms of chamber control (PID, bang-bang, and bang-bang with hysteresis).
 * When set to any value below 255, enables a form of PWM to the chamber heater that acts like a divider
 * so don't use it unless you are OK with PWM on your heater. (See the comment on enabling PIDTEMPCHAMBER)
 */
#define MAX_CHAMBER_POWER 255 // limits duty cycle to chamber heater; 255=full current
#if ENABLED(PIDTEMPCHAMBER)
  #define MIN_CHAMBER_POWER 0
  //#define PID_CHAMBER_DEBUG // Sends debug data to the serial port.
  // Lasko "MyHeat Personal Heater" (200w) modified with a Fotek SSR-10DA to control only the heating element
  // and placed inside the small Creality printer enclosure tent.
  //
  #define DEFAULT_chamberKp 37.04
  #define DEFAULT_chamberKi 1.40
  #define DEFAULT_chamberKd 655.17
  // M309 P37.04 I1.04 D655.17
  // FIND YOUR OWN: "M303 E-2 C8 S50" to run autotune on the chamber at 50 degreesC for 8 cycles.
#endif // PIDTEMPCHAMBER
#if ANY(PIDTEMP, PIDTEMPBED, PIDTEMPCHAMBER)
  //#define PID_DEBUG           // Sends debug data to the serial port. Use 'M303 D' to toggle activation.
  //#define PID_OPENLOOP        // Puts PID in open loop. M104/M140 sets the output power from 0 to PID_MAX
  //#define SLOW_PWM_HEATERS    // PWM with very low frequency (roughly 0.125Hz=8s) and minimum state time
                                // of approximately 1s useful for heaters driven by a relay
  #define PID_FUNCTIONAL_RANGE 10 // If the temperature difference between the target temperature and the actual
                                  // temperature
                                  // is more than PID_FUNCTIONAL_RANGE then the PID will be shut off and the
```

```
                                        // heater will be set to min/max.
#endif

// @section extruder
/**
 * Prevent extrusion if the temperature is below EXTRUDE_MINTEMP.
 * Add M302 to set the minimum extrusion temperature and/or turn
 * cold extrusion prevention on and off.
 *
 * *** IT IS HIGHLY RECOMMENDED TO LEAVE THIS OPTION ENABLED! ***
 */
#define PREVENT_COLD_EXTRUSION
#define EXTRUDE_MINTEMP 170
/**
 * Prevent a single extrusion longer than EXTRUDE_MAXLENGTH.
 * Note: For Bowden Extruders make this large enough to allow load/unload.
 */
#define PREVENT_LENGTHY_EXTRUDE
#define EXTRUDE_MAXLENGTH 200
//===========================================================================
//========================= Thermal Runaway Protection ======================
//===========================================================================
/**
 * Thermal Protection provides additional protection to your printer from damage
 * and fire. Marlin always includes safe min and max temperature ranges which
 * protect against a broken or disconnected thermistor wire.
 *
 * The issue: If a thermistor falls out, it will report the much lower
 * temperature of the air in the room, and the the firmware will keep
 * the heater on.
 *
 * If you get "Thermal Runaway" or "Heating failed" errors the
 * details can be tuned in Configuration_adv.h
 */
#define THERMAL_PROTECTION_HOTENDS // Enable thermal protection for all extruders
#define THERMAL_PROTECTION_BED     // Enable thermal protection for the heated bed
#define THERMAL_PROTECTION_CHAMBER // Enable thermal protection for the heated chamber
#define THERMAL_PROTECTION_COOLER  // Enable thermal protection for the laser cooling

//===========================================================================
//============================= Mechanical Settings =========================
//===========================================================================
// @section machine
// Enable one of the options below for CoreXY, CoreXZ, or CoreYZ kinematics,
// either in the usual order or reversed
//#define COREXY
//#define COREXZ
//#define COREYZ
//#define COREYX
//#define COREZX
//#define COREZY
//#define MARKFORGED_XY  // MarkForged. See https://reprap.org/forum/read.php?152,504042
// Enable for a belt style printer with endless "Z" motion
//#define BELTPRINTER

//===========================================================================
//============================= Endstop Settings ============================
//===========================================================================
// @section homing
// Specify here all the endstop connectors that are connected to any endstop or probe.
// Almost all printers will be using one per axis. Probes will use one or more of the
```

```
// extra connectors. Leave undefined any used for non-endstop and non-probe purposes.
#define USE_XMIN_PLUG
#define USE_YMIN_PLUG
#define USE_ZMIN_PLUG
//#define USE_IMIN_PLUG
//#define USE_JMIN_PLUG
//#define USE_KMIN_PLUG
//#define USE_XMAX_PLUG
//#define USE_YMAX_PLUG
//#define USE_ZMAX_PLUG
//#define USE_IMAX_PLUG
//#define USE_JMAX_PLUG
//#define USE_KMAX_PLUG
// Enable pullup for all endstops to prevent a floating state
#define ENDSTOPPULLUPS
#if DISABLED(ENDSTOPPULLUPS)
  // Disable ENDSTOPPULLUPS to set pullups individually
  //#define ENDSTOPPULLUP_XMAX
  //#define ENDSTOPPULLUP_YMAX
  //#define ENDSTOPPULLUP_ZMAX
  //#define ENDSTOPPULLUP_IMAX
  //#define ENDSTOPPULLUP_JMAX
  //#define ENDSTOPPULLUP_KMAX
  //#define ENDSTOPPULLUP_XMIN
  //#define ENDSTOPPULLUP_YMIN
  //#define ENDSTOPPULLUP_ZMIN
  //#define ENDSTOPPULLUP_IMIN
  //#define ENDSTOPPULLUP_JMIN
  //#define ENDSTOPPULLUP_KMIN
  //#define ENDSTOPPULLUP_ZMIN_PROBE
#endif
// Enable pulldown for all endstops to prevent a floating state
//#define ENDSTOPPULLDOWNS
#if DISABLED(ENDSTOPPULLDOWNS)
  // Disable ENDSTOPPULLDOWNS to set pulldowns individually
  //#define ENDSTOPPULLDOWN_XMAX
  //#define ENDSTOPPULLDOWN_YMAX
  //#define ENDSTOPPULLDOWN_ZMAX
  //#define ENDSTOPPULLDOWN_IMAX
  //#define ENDSTOPPULLDOWN_JMAX
  //#define ENDSTOPPULLDOWN_KMAX
  //#define ENDSTOPPULLDOWN_XMIN
  //#define ENDSTOPPULLDOWN_YMIN
  //#define ENDSTOPPULLDOWN_ZMIN
  //#define ENDSTOPPULLDOWN_IMIN
  //#define ENDSTOPPULLDOWN_JMIN
  //#define ENDSTOPPULLDOWN_KMIN
  //#define ENDSTOPPULLDOWN_ZMIN_PROBE
#endif
// Mechanical endstop with COM to ground and NC to Signal uses "false" here (most common setup).
#define X_MIN_ENDSTOP_INVERTING true // Set to true to invert the logic of the endstop.
#define Y_MIN_ENDSTOP_INVERTING true // Set to true to invert the logic of the endstop.
#define Z_MIN_ENDSTOP_INVERTING true // Set to true to invert the logic of the endstop.
#define I_MIN_ENDSTOP_INVERTING false // Set to true to invert the logic of the endstop.
#define J_MIN_ENDSTOP_INVERTING false // Set to true to invert the logic of the endstop.
#define K_MIN_ENDSTOP_INVERTING false // Set to true to invert the logic of the endstop.
#define X_MAX_ENDSTOP_INVERTING false // Set to true to invert the logic of the endstop.
#define Y_MAX_ENDSTOP_INVERTING false // Set to true to invert the logic of the endstop.
#define Z_MAX_ENDSTOP_INVERTING false // Set to true to invert the logic of the endstop.
#define I_MAX_ENDSTOP_INVERTING false // Set to true to invert the logic of the endstop.
#define J_MAX_ENDSTOP_INVERTING false // Set to true to invert the logic of the endstop.
```

```
#define K_MAX_ENDSTOP_INVERTING false // Set to true to invert the logic of the endstop.
#define Z_MIN_PROBE_ENDSTOP_INVERTING true // Set to true to invert the logic of the probe.
/**

 * Stepper Drivers
 *
 * These settings allow Marlin to tune stepper driver timing and enable advanced options for
 * stepper drivers that support them. You may also override timing options in Configuration_adv.h.
 *
 * A4988 is assumed for unspecified drivers.
 *
 * Use TMC2208/TMC2208_STANDALONE for TMC2225 drivers and TMC2209/TMC2209_STANDALONE for TMC2226
 * drivers.
 *
 * Options: A4988, A5984, DRV8825, LV8729, L6470, L6474, POWERSTEP01,
 *          TB6560, TB6600, TMC2100,
 *          TMC2130, TMC2130_STANDALONE, TMC2160, TMC2160_STANDALONE,
 *          TMC2208, TMC2208_STANDALONE, TMC2209, TMC2209_STANDALONE,
 *          TMC26X,  TMC26X_STANDALONE,  TMC2660, TMC2660_STANDALONE,
 *          TMC5130, TMC5130_STANDALONE, TMC5160, TMC5160_STANDALONE
 * :['A4988', 'A5984', 'DRV8825', 'LV8729', 'L6470', 'L6474', 'POWERSTEP01', 'TB6560', 'TB6600', 'TMC2100',
'TMC2130', 'TMC2130_STANDALONE', 'TMC2160', 'TMC2160_STANDALONE', 'TMC2208', 'TMC2208_STANDALONE',
'TMC2209', 'TMC2209_STANDALONE', 'TMC26X', 'TMC26X_STANDALONE', 'TMC2660', 'TMC2660_STANDALONE',
'TMC5130', 'TMC5130_STANDALONE', 'TMC5160', 'TMC5160_STANDALONE']
 */
#define X_DRIVER_TYPE  A4988
#define Y_DRIVER_TYPE  A4988
#define Z_DRIVER_TYPE  A4988
//#define X2_DRIVER_TYPE A4988
//#define Y2_DRIVER_TYPE A4988
//#define Z2_DRIVER_TYPE A4988
//#define Z3_DRIVER_TYPE A4988
//#define Z4_DRIVER_TYPE A4988
//#define I_DRIVER_TYPE  A4988
//#define J_DRIVER_TYPE  A4988
//#define K_DRIVER_TYPE  A4988
#define E0_DRIVER_TYPE A4988
//#define E1_DRIVER_TYPE A4988
//#define E2_DRIVER_TYPE A4988
//#define E3_DRIVER_TYPE A4988
//#define E4_DRIVER_TYPE A4988
//#define E5_DRIVER_TYPE A4988
//#define E6_DRIVER_TYPE A4988
//#define E7_DRIVER_TYPE A4988
// Enable this feature if all enabled endstop pins are interrupt-capable.
// This will remove the need to poll the interrupt pins, saving many CPU cycles.
//#define ENDSTOP_INTERRUPTS_FEATURE
/**

 * Endstop Noise Threshold
 *
 * Enable if your probe or endstops falsely trigger due to noise.
 *
 * - Higher values may affect repeatability or accuracy of some bed probes.
 * - To fix noise install a 100nF ceramic capacitor in parallel with the switch.
 * - This feature is not required for common micro-switches mounted on PCBs
 *   based on the Makerbot design, which already have the 100nF capacitor.
 *
 * :[2,3,4,5,6,7]
 */
//#define ENDSTOP_NOISE_THRESHOLD 2
```

```
// Check for stuck or disconnected endstops during homing moves.
//#define DETECT_BROKEN_ENDSTOP

//===========================================================================
//============================= Movement Settings ===========================
//===========================================================================
// @section motion
/**
 * Default Settings
 *
 * These settings can be reset by M502
 *
 * Note that if EEPROM is enabled, saved values will override these.
 */
/**
 * With this option each E stepper can have its own factors for the
 * following movement settings. If fewer factors are given than the
 * total number of extruders, the last value applies to the rest.
 */
//#define DISTINCT_E_FACTORS
/**
 * Default Axis Steps Per Unit (steps/mm)
 * Override with M92
 *                                  X, Y, Z [, I [, J [, K]]], E0 [, E1[, E2...]]
 */

// X, Y, Z, E = VIGTIGE PARAMETRE og er meget afhængig af, hvilke type steppermotorer der er benyttet.
// Brug evt. PRUSA CALCULATOR som nævnt øverst oppe i denne configuration.h
// Det er disse data, der bestemmer hvor meget en steppermotor flytter sig pr step.
#define DEFAULT_AXIS_STEPS_PER_UNIT   { 80, 80, 400, 65 }

/**
 * Default Max Feed Rate (mm/s)
 * Override with M203
 *                                  X, Y, Z [, I [, J [, K]]], E0 [, E1[, E2...]]
 */
#define DEFAULT_MAX_FEEDRATE          { 200, 200, 5, 25}
//#define LIMITED_MAX_FR_EDITING       // Limit edit via M203 or LCD to DEFAULT_MAX_FEEDRATE * 2
#if ENABLED(LIMITED_MAX_FR_EDITING)
  #define MAX_FEEDRATE_EDIT_VALUES    { 600, 600, 10, 50 } // ...or, set your own edit limits
#endif

/**
 * Default Max Acceleration (change/s) change = mm/s
 * (Maximum start speed for accelerated moves)
 * Override with M201
 *                                  X, Y, Z [, I [, J [, K]]], E0 [, E1[, E2...]]
 */
#define DEFAULT_MAX_ACCELERATION      { 200,200,50,1000 }
//#define LIMITED_MAX_ACCEL_EDITING     // Limit edit via M201 or LCD to DEFAULT_MAX_ACCELERATION * 2
#if ENABLED(LIMITED_MAX_ACCEL_EDITING)
  #define MAX_ACCEL_EDIT_VALUES       { 600, 600, 200, 20000 } // ...or, set your own edit limits
#endif
/**
 * Default Acceleration (change/s) change = mm/s
 * Override with M204
 *
 *   M204 P    Acceleration
 *   M204 R    Retract Acceleration
 *   M204 T    Travel Acceleration
 */
```

```
#define DEFAULT_ACCELERATION          100    // X, Y, Z and E acceleration for printing moves
#define DEFAULT_RETRACT_ACCELERATION  100    // E acceleration for retracts
#define DEFAULT_TRAVEL_ACCELERATION   100    // X, Y, Z acceleration for travel (non printing) moves
/**
 * Default Jerk limits (mm/s)
 * Override with M205 X Y Z E
 *
 * "Jerk" specifies the minimum speed change that requires acceleration.
 * When changing speed and direction, if the difference is less than the
 * value set here, it may happen instantaneously.
 */
#define CLASSIC_JERK
#if ENABLED(CLASSIC_JERK)
  #define DEFAULT_XJERK 5.0
  #define DEFAULT_YJERK 5.0
  #define DEFAULT_ZJERK  0.4
  //#define DEFAULT_IJERK  0.3
  //#define DEFAULT_JJERK  0.3
  //#define DEFAULT_KJERK  0.3
  //#define TRAVEL_EXTRA_XYJERK 0.0     // Additional jerk allowance for all travel moves
  //#define LIMITED_JERK_EDITING        // Limit edit via M205 or LCD to DEFAULT_aJERK * 2
  #if ENABLED(LIMITED_JERK_EDITING)
    #define MAX_JERK_EDIT_VALUES { 20, 20, 0.6, 10 } // ...or, set your own edit limits
  #endif
#endif
#define DEFAULT_EJERK    5.0  // May be used by Linear Advance
/**
 * Junction Deviation Factor
 *
 * See:
 *   https://reprap.org/forum/read.php?1,739819
 *   https://blog.kyneticcnc.com/2018/10/computing-junction-deviation-for-marlin.html
 */
#if DISABLED(CLASSIC_JERK)
  #define JUNCTION_DEVIATION_MM 0.013 // (mm) Distance from real junction edge
  #define JD_HANDLE_SMALL_SEGMENTS    // Use curvature estimation instead of just the junction angle
                                      // for small segments (< 1mm) with large junction angles (> 135°).
#endif

/**
 * S-Curve Acceleration
 *
 * This option eliminates vibration during printing by fitting a Bézier
 * curve to move acceleration, producing much smoother direction changes.
 *
 * See https://github.com/synthetos/TinyG/wiki/Jerk-Controlled-Motion-Explained
 */
#define S_CURVE_ACCELERATION

//===========================================================================
//============================= Z Probe Options =============================
//===========================================================================
// @section probes
//
// See https://marlinfw.org/docs/configuration/probes.html
//
/**
 * Enable this option for a probe connected to the Z-MIN pin.
 * The probe replaces the Z-MIN endstop and is used for Z homing.
 * (Automatically enables USE_PROBE_FOR_Z_HOMING.)
 */
```

```
#define Z_MIN_PROBE_USES_Z_MIN_ENDSTOP_PIN
// Force the use of the probe for Z-axis homing
//#define USE_PROBE_FOR_Z_HOMING
/**
 * Z_MIN_PROBE_PIN
 *
 * Define this pin if the probe is not connected to Z_MIN_PIN.
 * If not defined the default pin for the selected MOTHERBOARD
 * will be used. Most of the time the default is what you want.
 *
 *  - The simplest option is to use a free endstop connector.
 *  - Use 5V for powered (usually inductive) sensors.
 *
 *  - RAMPS 1.3/1.4 boards may use the 5V, GND, and Aux4->D32 pin:
 *    - For simple switches connect...
 *      - normally-closed switches to GND and D32.
 *      - normally-open switches to 5V and D32.
 */
//#define Z_MIN_PROBE_PIN 32 // Pin 32 is the RAMPS default
/**
 * Probe Type
 *
 * Allen Key Probes, Servo Probes, Z-Sled Probes, FIX_MOUNTED_PROBE, etc.
 * Activate one of these to use Auto Bed Leveling below.
 */
/**
 * The "Manual Probe" provides a means to do "Auto" Bed Leveling without a probe.
 * Use G29 repeatedly, adjusting the Z height at each point with movement commands
 * or (with LCD_BED_LEVELING) the LCD controller.
 */
//#define PROBE_MANUALLY
/**
 * A Fix-Mounted Probe either doesn't deploy or needs manual deployment.
 *   (e.g., an inductive probe or a nozzle-based probe-switch.)
 */
#define FIX_MOUNTED_PROBE    // Min probe skal have 12 volt for at virke
                             // se: https://planker.dk/Projects/Arduino/3D-print/Softwareinstallation.htm

/**
 * Use the nozzle as the probe, as with a conductive
 * nozzle system or a piezo-electric smart effector.
 */
//#define NOZZLE_AS_PROBE

/**
 * Z Servo Probe, such as an endstop switch on a rotating arm.
 */
//#define Z_PROBE_SERVO_NR 0       // Defaults to SERVO 0 connector.
//#define Z_SERVO_ANGLES { 70, 0 } // Z Servo Deploy and Stow angles
/**
 * The BLTouch probe uses a Hall effect sensor and emulates a servo.
 */
//#define BLTOUCH
/**
 * Touch-MI Probe by hotends.fr
 *
 * This probe is deployed and activated by moving the X-axis to a magnet at the edge of the bed.
 * By default, the magnet is assumed to be on the left and activated by a home. If the magnet is
 * on the right, enable and set TOUCH_MI_DEPLOY_XPOS to the deploy position.
 *
 * Also requires: BABYSTEPPING, BABYSTEP_ZPROBE_OFFSET, Z_SAFE_HOMING,
```

```
 *             and a minimum Z_HOMING_HEIGHT of 10.
 */
//#define TOUCH_MI_PROBE
#if ENABLED(TOUCH_MI_PROBE)
  #define TOUCH_MI_RETRACT_Z 0.5               // Height at which the probe retracts
  //#define TOUCH_MI_DEPLOY_XPOS (X_MAX_BED + 2)  // For a magnet on the right side of the bed
  //#define TOUCH_MI_MANUAL_DEPLOY               // For manual deploy (LCD menu)
#endif
// A probe that is deployed and stowed with a solenoid pin (SOL1_PIN)
//#define SOLENOID_PROBE
// A sled-mounted probe like those designed by Charles Bell.
//#define Z_PROBE_SLED
//#define SLED_DOCKING_OFFSET 5  // The extra distance the X axis must travel to pickup the sled. 0 should be fine
                                 // but you can push it further if you'd like.
// A probe deployed by moving the x-axis, such as the Wilson II's rack-and-pinion probe designed by Marty Rice.
//#define RACK_AND_PINION_PROBE
#if ENABLED(RACK_AND_PINION_PROBE)
  #define Z_PROBE_DEPLOY_X  X_MIN_POS
  #define Z_PROBE_RETRACT_X X_MAX_POS
#endif
// Duet Smart Effector (for delta printers) - https://bit.ly/2ul5U7J
// When the pin is defined you can use M672 to set/reset the probe sensitivity.
//#define DUET_SMART_EFFECTOR
#if ENABLED(DUET_SMART_EFFECTOR)
  #define SMART_EFFECTOR_MOD_PIN  -1  // Connect a GPIO pin to the Smart Effector MOD pin
#endif
/**
 * Use StallGuard2 to probe the bed with the nozzle.
 * Requires stallGuard-capable Trinamic stepper drivers.
 * CAUTION: This can damage machines with Z lead screws.
 *          Take extreme care when setting up this feature.
 */
//#define SENSORLESS_PROBING
//
// For Z_PROBE_ALLEN_KEY see the Delta example configurations.
//
/**
 * Nozzle-to-Probe offsets { X, Y, Z }
 *
 * X and Y offset
 *   Use a caliper or ruler to measure the distance from the tip of
 *   the Nozzle to the center-point of the Probe in the X and Y axes.
 *
 * Z offset
 * - For the Z offset use your best known value and adjust at runtime.
 * - Common probes trigger below the nozzle and have negative values for Z offset.
 * - Probes triggering above the nozzle height are uncommon but do exist. When using
 *   probes such as this, carefully set Z_CLEARANCE_DEPLOY_PROBE and Z_CLEARANCE_BETWEEN_PROBES
 *   to avoid collisions during probing.
 *
 * Tune and Adjust
 * -  Probe Offsets can be tuned at runtime with 'M851', LCD menus, babystepping, etc.
 * -  PROBE_OFFSET_WIZARD (configuration_adv.h) can be used for setting the Z offset.
 *
 * Assuming the typical work area orientation:
 *  - Probe to RIGHT of the Nozzle has a Positive X offset
 *  - Probe to LEFT  of the Nozzle has a Negative X offset
 *  - Probe in BACK  of the Nozzle has a Positive Y offset
 *  - Probe in FRONT of the Nozzle has a Negative Y offset
 *
 * Some examples:
```

```
 *   #define NOZZLE_TO_PROBE_OFFSET { 10, 10, -1 }   // Example "1"
 *   #define NOZZLE_TO_PROBE_OFFSET {-10,  5, -1 }   // Example "2"
 *   #define NOZZLE_TO_PROBE_OFFSET {  5, -5, -1 }   // Example "3"
 *   #define NOZZLE_TO_PROBE_OFFSET {-15,-10, -1 }   // Example "4"
 *
 *      +-- BACK ---+
 *      |    [+]    |
 *    L |        1  | R <-- Example "1" (right+,  back+)
 *    E |  2        | I <-- Example "2" ( left-,  back+)
 *    F |[-]  N  [+]| G <-- Nozzle
 *    T |       3   | H <-- Example "3" (right+, front-)
 *      | 4         | T <-- Example "4" ( left-, front-)
 *      |    [-]    |
 *      O-- FRONT --+
 */
#define NOZZLE_TO_PROBE_OFFSET { -20, 0, 0 }   // X= -20, Y= 0, Z= 0

// Most probes should stay away from the edges of the bed, but
// with NOZZLE_AS_PROBE this can be negative for a wider probing area.
#define PROBING_MARGIN 10
// X and Y axis travel speed (mm/min) between probes
#define XY_PROBE_FEEDRATE (50*60) // was: 133
// Feedrate (mm/min) for the first approach when double-probing (MULTIPLE_PROBING == 2)
#define Z_PROBE_FEEDRATE_FAST (4*60)
// Feedrate (mm/min) for the "accurate" probe of each point
#define Z_PROBE_FEEDRATE_SLOW (Z_PROBE_FEEDRATE_FAST / 2)
/**
 * Probe Activation Switch
 * A switch indicating proper deployment, or an optical
 * switch triggered when the carriage is near the bed.
 */
//#define PROBE_ACTIVATION_SWITCH
#if ENABLED(PROBE_ACTIVATION_SWITCH)
  #define PROBE_ACTIVATION_SWITCH_STATE LOW // State indicating probe is active
  //#define PROBE_ACTIVATION_SWITCH_PIN PC6 // Override default pin
#endif
/**
 * Tare Probe (determine zero-point) prior to each probe.
 * Useful for a strain gauge or piezo sensor that needs to factor out
 * elements such as cables pulling on the carriage.
 */
//#define PROBE_TARE
#if ENABLED(PROBE_TARE)
  #define PROBE_TARE_TIME  200    // (ms) Time to hold tare pin
  #define PROBE_TARE_DELAY 200    // (ms) Delay after tare before
  #define PROBE_TARE_STATE HIGH   // State to write pin for tare
  //#define PROBE_TARE_PIN PA5    // Override default pin
  #if ENABLED(PROBE_ACTIVATION_SWITCH)
    //#define PROBE_TARE_ONLY_WHILE_INACTIVE  // Fail to tare/probe if PROBE_ACTIVATION_SWITCH is active
  #endif
#endif
/**
 * Multiple Probing
 *
 * You may get improved results by probing 2 or more times.
 * With EXTRA_PROBING the more atypical reading(s) will be disregarded.
 *
 * A total of 2 does fast/slow probes with a weighted average.
 * A total of 3 or more adds more slow probes, taking the average.
 */
#define MULTIPLE_PROBING 2
#define EXTRA_PROBING    1
```

```
/**
 * Z probes require clearance when deploying, stowing, and moving between
 * probe points to avoid hitting the bed and other hardware.
 * Servo-mounted probes require extra space for the arm to rotate.
 * Inductive probes need space to keep from triggering early.
 *
 * Use these settings to specify the distance (mm) to raise the probe (or
 * lower the bed). The values set here apply over and above any (negative)
 * probe Z Offset set with NOZZLE_TO_PROBE_OFFSET, M851, or the LCD.
 * Only integer values >= 1 are valid here.
 *
 * Example: `M851 Z-5` with a CLEARANCE of 4  =>  9mm from bed to nozzle.
 *     But: `M851 Z+1` with a CLEARANCE of 2  =>  2mm from bed to nozzle.
 */
#define Z_CLEARANCE_DEPLOY_PROBE     10 // Z Clearance for Deploy/Stow
#define Z_CLEARANCE_BETWEEN_PROBES  5 // Z Clearance between probe points
#define Z_CLEARANCE_MULTI_PROBE      5 // Z Clearance between multiple probes
//#define Z_AFTER_PROBING              5 // Z position after probing is done
#define Z_PROBE_LOW_POINT           -2 // Farthest distance below the trigger-point to go before stopping
// For M851 give a range for adjusting the Z probe offset
#define Z_PROBE_OFFSET_RANGE_MIN -20
#define Z_PROBE_OFFSET_RANGE_MAX 20
// Enable the M48 repeatability test to test probe accuracy
//#define Z_MIN_PROBE_REPEATABILITY_TEST
// Before deploy/stow pause for user confirmation
//#define PAUSE_BEFORE_DEPLOY_STOW
#if ENABLED(PAUSE_BEFORE_DEPLOY_STOW)
  //#define PAUSE_PROBE_DEPLOY_WHEN_TRIGGERED // For Manual Deploy Allenkey Probe
#endif

/**
 * Enable one or more of the following if probing seems unreliable.
 * Heaters and/or fans can be disabled during probing to minimize electrical
 * noise. A delay can also be added to allow noise and vibration to settle.
 * These options are most useful for the BLTouch probe, but may also improve
 * readings with inductive probes and piezo sensors.
 */
//#define PROBING_HEATERS_OFF       // Turn heaters off when probing
#if ENABLED(PROBING_HEATERS_OFF)
  //#define WAIT_FOR_BED_HEATER     // Wait for bed to heat back up between probes (to improve accuracy)
  //#define WAIT_FOR_HOTEND         // Wait for hotend to heat back up between probes (to improve accuracy &
                                    // revent cold extrude)

#endif
//#define PROBING_FANS_OFF          // Turn fans off when probing
//#define PROBING_ESTEPPERS_OFF     // Turn all extruder steppers off when probing
//#define PROBING_STEPPERS_OFF      // Turn all steppers off (unless needed to hold position) when probing
                                    // (including extruders)
//#define DELAY_BEFORE_PROBING 200  // (ms) To prevent vibrations from triggering piezo sensors
// Require minimum nozzle and/or bed temperature for probing
//#define PREHEAT_BEFORE_PROBING
#if ENABLED(PREHEAT_BEFORE_PROBING)
  #define PROBING_NOZZLE_TEMP 120   // (°C) Only applies to E0 at this time
  #define PROBING_BED_TEMP     50
#endif
// For Inverting Stepper Enable Pins (Active Low) use 0, Non Inverting (Active High) use 1
// :{ 0:'Low', 1:'High' }
#define X_ENABLE_ON 0
#define Y_ENABLE_ON 0
#define Z_ENABLE_ON 0
#define E_ENABLE_ON 0 // For all extruders
```

```
//#define I_ENABLE_ON 0
//#define J_ENABLE_ON 0
//#define K_ENABLE_ON 0
// Disable axis steppers immediately when they're not being stepped.
// WARNING: When motors turn off there is a chance of losing position accuracy!
#define DISABLE_X false
#define DISABLE_Y false
#define DISABLE_Z false
//#define DISABLE_I false
//#define DISABLE_J false
//#define DISABLE_K false
// Turn off the display blinking that warns about possible accuracy reduction
//#define DISABLE_REDUCED_ACCURACY_WARNING
// @section extruder
#define DISABLE_E false           // Disable the extruder when not stepping
#define DISABLE_INACTIVE_EXTRUDER   // Keep only the active extruder enabled

// @section machine
// Invert the stepper direction. Change (or reverse the motor connector) if an axis goes the wrong way.
#define INVERT_X_DIR false
#define INVERT_Y_DIR false
#define INVERT_Z_DIR false
//#define INVERT_I_DIR false
//#define INVERT_J_DIR false
//#define INVERT_K_DIR false
// @section extruder
// For direct drive extruder v9 set to true, for geared extruder set to false.
#define INVERT_E0_DIR false
//#define INVERT_E1_DIR false
//#define INVERT_E2_DIR false
//#define INVERT_E3_DIR false
//#define INVERT_E4_DIR false
//#define INVERT_E5_DIR false
//#define INVERT_E6_DIR false
//#define INVERT_E7_DIR false

// @section homing
#define NO_MOTION_BEFORE_HOMING // Inhibit movement until all axes have been homed. Also enable
                                // HOME_AFTER_DEACTIVATE for extra safety.
#define HOME_AFTER_DEACTIVATE     // Require rehoming after steppers are deactivated. Also enable
                                // NO_MOTION_BEFORE_HOMING for extra safety.
/**
 * Set Z_IDLE_HEIGHT if the Z-Axis moves on its own when steppers are disabled.
 *  - Use a low value (i.e., Z_MIN_POS) if the nozzle falls down to the bed.
 *  - Use a large value (i.e., Z_MAX_POS) if the bed falls down, away from the nozzle.
 */
//#define Z_IDLE_HEIGHT Z_HOME_POS
#define Z_HOMING_HEIGHT  4  // (mm) Minimal Z height before homing (G28) for Z clearance above the bed,
                            // clamps, ...
                            // Be sure to have this much clearance over your Z_MAX_POS to prevent grinding.
#define Z_AFTER_HOMING  10     // (mm) Height to move to after homing Z
// Direction of endstops when homing; 1=MAX, -1=MIN
// :[-1,1]
#define X_HOME_DIR -1
#define Y_HOME_DIR -1
#define Z_HOME_DIR -1
//#define I_HOME_DIR -1
//#define J_HOME_DIR -1
//#define K_HOME_DIR -1
```

```
// @section machine
// The size of the printable area
#define X_BED_SIZE 155
#define Y_BED_SIZE 170
// Travel limits (mm) after homing, corresponding to endstop positions.
#define X_MIN_POS 0
#define Y_MIN_POS 0
#define Z_MIN_POS 0
#define X_MAX_POS X_BED_SIZE
#define Y_MAX_POS Y_BED_SIZE
#define Z_MAX_POS 140
//#define I_MIN_POS 0
//#define I_MAX_POS 50
//#define J_MIN_POS 0
//#define J_MAX_POS 50
//#define K_MIN_POS 0
//#define K_MAX_POS 50

/**
 * Software Endstops
 *
 * - Prevent moves outside the set machine bounds.
 * - Individual axes can be disabled, if desired.
 * - X and Y only apply to Cartesian robots.
 * - Use 'M211' to set software endstops on/off or report current state
 */
// Min software endstops constrain movement within minimum coordinate bounds
#define MIN_SOFTWARE_ENDSTOPS
#if ENABLED(MIN_SOFTWARE_ENDSTOPS)
  #define MIN_SOFTWARE_ENDSTOP_X
  #define MIN_SOFTWARE_ENDSTOP_Y
  #define MIN_SOFTWARE_ENDSTOP_Z
  #define MIN_SOFTWARE_ENDSTOP_I
  #define MIN_SOFTWARE_ENDSTOP_J
  #define MIN_SOFTWARE_ENDSTOP_K
#endif
// Max software endstops constrain movement within maximum coordinate bounds
#define MAX_SOFTWARE_ENDSTOPS
#if ENABLED(MAX_SOFTWARE_ENDSTOPS)
  #define MAX_SOFTWARE_ENDSTOP_X
  #define MAX_SOFTWARE_ENDSTOP_Y
  #define MAX_SOFTWARE_ENDSTOP_Z
  #define MAX_SOFTWARE_ENDSTOP_I
  #define MAX_SOFTWARE_ENDSTOP_J
  #define MAX_SOFTWARE_ENDSTOP_K
#endif
#if EITHER(MIN_SOFTWARE_ENDSTOPS, MAX_SOFTWARE_ENDSTOPS)
  //#define SOFT_ENDSTOPS_MENU_ITEM  // Enable/Disable software endstops from the LCD
#endif

/**
 * Filament Runout Sensors
 * Mechanical or opto endstops are used to check for the presence of filament.
 *
 * IMPORTANT: Runout will only trigger if Marlin is aware that a print job is running.
 * Marlin knows a print job is running when:
 *  1. Running a print job from media started with M24.
 *  2. The Print Job Timer has been started with M75.
 *  3. The heaters were turned on and PRINTJOB_TIMER_AUTOSTART is enabled.
 *
 * RAMPS-based boards use SERVO3_PIN for the first runout sensor.
```

```
 * For other boards you may need to define FIL_RUNOUT_PIN, FIL_RUNOUT2_PIN, etc.
 */
#define FILAMENT_RUNOUT_SENSOR
#if ENABLED(FILAMENT_RUNOUT_SENSOR)
  #define FIL_RUNOUT_ENABLED_DEFAULT true // Enable the sensor on startup. Override with M412 followed by
                                          // 500.
  #define NUM_RUNOUT_SENSORS   1          // Number of sensors, up to one per extruder.
                                          // Define a FIL_RUNOUT#_PIN for each.
  #define FIL_RUNOUT_STATE     LOW        // Pin state indicating that filament is NOT present.
  #define FIL_RUNOUT_PULLUP               // Use internal pullup for filament runout pins.
  //#define FIL_RUNOUT_PULLDOWN           // Use internal pulldown for filament runout pins.
  //#define WATCH_ALL_RUNOUT_SENSORS      // Execute runout script on any triggering sensor, not only for the
                                          // active extruder.
                                // This is automatically enabled for MIXING_EXTRUDERs.
  // Override individually if the runout sensors vary
  //#define FIL_RUNOUT1_STATE LOW
  //#define FIL_RUNOUT1_PULLUP
  //#define FIL_RUNOUT1_PULLDOWN
  //#define FIL_RUNOUT2_STATE LOW
  //#define FIL_RUNOUT2_PULLUP
  //#define FIL_RUNOUT2_PULLDOWN
  //#define FIL_RUNOUT3_STATE LOW
  //#define FIL_RUNOUT3_PULLUP
  //#define FIL_RUNOUT3_PULLDOWN
  //#define FIL_RUNOUT4_STATE LOW
  //#define FIL_RUNOUT4_PULLUP
  //#define FIL_RUNOUT4_PULLDOWN
  //#define FIL_RUNOUT5_STATE LOW
  //#define FIL_RUNOUT5_PULLUP
  //#define FIL_RUNOUT5_PULLDOWN
  //#define FIL_RUNOUT6_STATE LOW
  //#define FIL_RUNOUT6_PULLUP
  //#define FIL_RUNOUT6_PULLDOWN
  //#define FIL_RUNOUT7_STATE LOW
  //#define FIL_RUNOUT7_PULLUP
  //#define FIL_RUNOUT7_PULLDOWN
  //#define FIL_RUNOUT8_STATE LOW
  //#define FIL_RUNOUT8_PULLUP
  //#define FIL_RUNOUT8_PULLDOWN
  // Commands to execute on filament runout.
  // With multiple runout sensors use the %c placeholder for the current tool in commands (e.g., "M600 T%c")
  // NOTE: After 'M412 H1' the host handles filament runout and this script does not apply.
  #define FILAMENT_RUNOUT_SCRIPT "M600"
  // After a runout is detected, continue printing this length of filament
  // before executing the runout script. Useful for a sensor at the end of
  // a feed tube. Requires 4 bytes SRAM per sensor, plus 4 bytes overhead.
  #define FILAMENT_RUNOUT_DISTANCE_MM 450
  #ifdef FILAMENT_RUNOUT_DISTANCE_MM
    // Enable this option to use an encoder disc that toggles the runout pin
    // as the filament moves. (Be sure to set FILAMENT_RUNOUT_DISTANCE_MM
    // large enough to avoid false positives.)
    //#define FILAMENT_MOTION_SENSOR
  #endif
#endif

//===========================================================================
//============================= Bed Leveling ============================
//===========================================================================
// @section calibrate
/**
 * Choose one of the options below to enable G29 Bed Leveling.
```

```
 * The parameters and behavior of G29 will change depending on your selection.
 *
 *  If using a Probe for Z Homing, enable Z_SAFE_HOMING also!
 *
 * - AUTO_BED_LEVELING_3POINT
 *   Probe 3 arbitrary points on the bed (that aren't collinear)
 *   You specify the XY coordinates of all 3 points.
 *   The result is a single tilted plane. Best for a flat bed.
 *
 * - AUTO_BED_LEVELING_LINEAR
 *   Probe several points in a grid.
 *   You specify the rectangle and the density of sample points.
 *   The result is a single tilted plane. Best for a flat bed.
 *
 * - AUTO_BED_LEVELING_BILINEAR
 *   Probe several points in a grid.
 *   You specify the rectangle and the density of sample points.
 *   The result is a mesh, best for large or uneven beds.
 *
 * - AUTO_BED_LEVELING_UBL (Unified Bed Leveling)
 *   A comprehensive bed leveling system combining the features and benefits
 *   of other systems. UBL also includes integrated Mesh Generation, Mesh
 *   Validation and Mesh Editing systems.
 *
 * - MESH_BED_LEVELING
 *   Probe a grid manually
 *   The result is a mesh, suitable for large or uneven beds. (See BILINEAR.)
 *   For machines without a probe, Mesh Bed Leveling provides a method to perform
 *   leveling in steps so you can manually adjust the Z height at each grid-point.
 *   With an LCD controller the process is guided step-by-step.
 */
//#define AUTO_BED_LEVELING_3POINT
//#define AUTO_BED_LEVELING_LINEAR
#define AUTO_BED_LEVELING_BILINEAR
//#define AUTO_BED_LEVELING_UBL
//#define MESH_BED_LEVELING
/**
 * Normally G28 leaves leveling disabled on completion. Enable one of
 * these options to restore the prior leveling state or to always enable
 * leveling immediately after G28.
 */
#define RESTORE_LEVELING_AFTER_G28
//#define ENABLE_LEVELING_AFTER_G28
/**
 * Auto-leveling needs preheating
 */
//#define PREHEAT_BEFORE_LEVELING
#if ENABLED(PREHEAT_BEFORE_LEVELING)
  #define LEVELING_NOZZLE_TEMP 120   // (°C) Only applies to E0 at this time
  #define LEVELING_BED_TEMP     50
#endif
/**
 * Enable detailed logging of G28, G29, M48, etc.
 * Turn on with the command 'M111 S32'.
 * NOTE: Requires a lot of PROGMEM!
 */
//#define DEBUG_LEVELING_FEATURE
#if ANY(MESH_BED_LEVELING, AUTO_BED_LEVELING_UBL, PROBE_MANUALLY)
  // Set a height for the start of manual adjustment
  #define MANUAL_PROBE_START_Z 0.2  // (mm) Comment out to use the last-measured height
#endif
```

```
#if ANY(MESH_BED_LEVELING, AUTO_BED_LEVELING_BILINEAR, AUTO_BED_LEVELING_UBL)
  // Gradually reduce leveling correction until a set height is reached,
  // at which point movement will be level to the machine's XY plane.
  // The height can be set with M420 Z<height>
  #define ENABLE_LEVELING_FADE_HEIGHT
  #if ENABLED(ENABLE_LEVELING_FADE_HEIGHT)
    #define DEFAULT_LEVELING_FADE_HEIGHT 10.0 // (mm) Default fade height.
  #endif
  // For Cartesian machines, instead of dividing moves on mesh boundaries,
  // split up moves into short segments like a Delta. This follows the
  // contours of the bed more closely than edge-to-edge straight moves.
  #define SEGMENT_LEVELED_MOVES
  #define LEVELED_SEGMENT_LENGTH 5.0 // (mm) Length of all segments (except the last one)
  /**
   * Enable the G26 Mesh Validation Pattern tool.
   */
  //#define G26_MESH_VALIDATION
  #if ENABLED(G26_MESH_VALIDATION)
    #define MESH_TEST_NOZZLE_SIZE    0.4  // (mm) Diameter of primary nozzle.
    #define MESH_TEST_LAYER_HEIGHT   0.2  // (mm) Default layer height for G26.
    #define MESH_TEST_HOTEND_TEMP  205    // (°C) Default nozzle temperature for G26.
    #define MESH_TEST_BED_TEMP      60    // (°C) Default bed temperature for G26.
    #define G26_XY_FEEDRATE         20    // (mm/s) Feedrate for G26 XY moves.
    #define G26_XY_FEEDRATE_TRAVEL 100    // (mm/s) Feedrate for G26 XY travel moves.
    #define G26_RETRACT_MULTIPLIER   1.0  // G26 Q (retraction) used by default between mesh test elements.
  #endif
#endif
#if EITHER(AUTO_BED_LEVELING_LINEAR, AUTO_BED_LEVELING_BILINEAR)
  // Set the number of grid points per dimension.
  #define GRID_MAX_POINTS_X 3
  #define GRID_MAX_POINTS_Y GRID_MAX_POINTS_X
  // Probe along the Y axis, advancing X after each column
  //#define PROBE_Y_FIRST
  #if ENABLED(AUTO_BED_LEVELING_BILINEAR)
    // Beyond the probed grid, continue the implied tilt?
    // Default is to maintain the height of the nearest edge.
    //#define EXTRAPOLATE_BEYOND_GRID
    //
    // Experimental Subdivision of the grid by Catmull-Rom method.
    // Synthesizes intermediate points to produce a more detailed mesh.
    //
    #define ABL_BILINEAR_SUBDIVISION
    #if ENABLED(ABL_BILINEAR_SUBDIVISION)
      // Number of subdivisions between probe points
      #define BILINEAR_SUBDIVISIONS 3
    #endif
  #endif
#elif ENABLED(AUTO_BED_LEVELING_UBL)
//===========================================================================
  //========================== Unified Bed Leveling ============================
//===========================================================================
  //#define MESH_EDIT_GFX_OVERLAY   // Display a graphics overlay while editing the mesh
  #define MESH_INSET 1              // Set Mesh bounds as an inset region of the bed
  #define GRID_MAX_POINTS_X 10      // Don't use more than 15 points per axis, implementation limited.
  #define GRID_MAX_POINTS_Y GRID_MAX_POINTS_X
  //#define UBL_HILBERT_CURVE       // Use Hilbert distribution for less travel when probing multiple points
  #define UBL_MESH_EDIT_MOVES_Z     // Sophisticated users prefer no movement of nozzle
  #define UBL_SAVE_ACTIVE_ON_M500   // Save the currently active mesh in the current slot on M500
  //#define UBL_Z_RAISE_WHEN_OFF_MESH 2.5 // When the nozzle is off the mesh, this value is used
                                          // as the Z-Height correction value.
  //#define UBL_MESH_WIZARD         // Run several commands in a row to get a complete mesh
```

```
#elif ENABLED(MESH_BED_LEVELING)
//===========================================================================
  //=================================== Mesh ================================
//===========================================================================
  #define MESH_INSET 10         // Set Mesh bounds as an inset region of the bed
  #define GRID_MAX_POINTS_X 2    // Don't use more than 7 points per axis, implementation limited.
  #define GRID_MAX_POINTS_Y GRID_MAX_POINTS_X

  //#define MESH_G28_REST_ORIGIN // After homing all axes ('G28' or 'G28 XYZ') rest Z at Z_MIN_POS
#endif // BED_LEVELING
/**
 * Add a bed leveling sub-menu for ABL or MBL.
 * Include a guided procedure if manual probing is enabled.
 */
//#define LCD_BED_LEVELING
#if ENABLED(LCD_BED_LEVELING)
  #define MESH_EDIT_Z_STEP  0.025 // (mm) Step size while manually probing Z axis.
  #define LCD_PROBE_Z_RANGE 4     // (mm) Z Range centered on Z_MIN_POS for LCD Z adjustment
  //#define MESH_EDIT_MENU        // Add a menu to edit mesh points
#endif
// Add a menu item to move between bed corners for manual bed adjustment
//#define LEVEL_BED_CORNERS
#if ENABLED(LEVEL_BED_CORNERS)
  #define LEVEL_CORNERS_INSET_LFRB { 30, 30, 30, 30 } // (mm) Left, Front, Right, Back insets
  #define LEVEL_CORNERS_HEIGHT      0.0   // (mm) Z height of nozzle at leveling points
  #define LEVEL_CORNERS_Z_HOP       4.0   // (mm) Z height of nozzle between leveling points
  //#define LEVEL_CENTER_TOO            // Move to the center after the last corner
  //#define LEVEL_CORNERS_USE_PROBE
  #if ENABLED(LEVEL_CORNERS_USE_PROBE)
    #define LEVEL_CORNERS_PROBE_TOLERANCE 0.1
    #define LEVEL_CORNERS_VERIFY_RAISED   // After adjustment triggers the probe, re-probe to verify
    //#define LEVEL_CORNERS_AUDIO_FEEDBACK
  #endif


/**
   * Corner Leveling Order
   *
   * Set 2 or 4 points. When 2 points are given, the 3rd is the center of the opposite edge.
   *
   *  LF  Left-Front    RF  Right-Front
   *  LB  Left-Back     RB  Right-Back
   *
   * Examples:
   *
    *       Default         {LF,RB,LB,RF}          {LF,RF}            {LB,LF}
    *  LB --------- RB    LB --------- RB     LB --------- RB    LB --------- RB
    *  |  4        3  |    | 3         2 |     |      <3>      |    | 1           |
    *  |              |    |             |     |               |    |          <3>|
    *  |  1        2  |    | 1         4 |     | 1          2  |    | 2           |
    *  LF --------- RF    LF --------- RF     LF --------- RF    LF --------- RF
   */
   #define LEVEL_CORNERS_LEVELING_ORDER { LF, RF, RB, LB } // Aktivated
#endif


/**
 * Commands to execute at the end of G29 probing.
 * Useful to retract or move the Z probe out of the way.
 */
//#define Z_PROBE_END_SCRIPT "G1 Z10 F12000\nG1 X15 Y330\nG1 Z0.5\nG1 Z10"
```

```
// @section homing
// The center of the bed is at (X=0, Y=0)
//#define BED_CENTER_AT_0_0
// Manually set the home position. Leave these undefined for automatic settings.
// For DELTA this is the top-center of the Cartesian print volume.
//#define MANUAL_X_HOME_POS 0
//#define MANUAL_Y_HOME_POS 0
//#define MANUAL_Z_HOME_POS 0
//#define MANUAL_I_HOME_POS 0
//#define MANUAL_J_HOME_POS 0
//#define MANUAL_K_HOME_POS 0
/**
 * Use "Z Safe Homing" to avoid homing with a Z probe outside the bed area.
 *
 * - Moves the Z probe (or nozzle) to a defined XY point before Z homing.
 * - Allows Z homing only when XY positions are known and trusted.
 * - If stepper drivers sleep, XY homing may be required again before Z homing.
 */
#define Z_SAFE_HOMING
#if ENABLED(Z_SAFE_HOMING)
  #define Z_SAFE_HOMING_X_POINT X_CENTER  // X point for Z homing
  #define Z_SAFE_HOMING_Y_POINT Y_CENTER  // Y point for Z homing
#endif
// Homing speeds (mm/min)
#define HOMING_FEEDRATE_MM_M { (50*60), (50*60), (4*60) }
// Validate that endstops are triggered on homing moves
#define VALIDATE_HOMING_ENDSTOPS
// @section calibrate
/**
 * Bed Skew Compensation
 *
 * This feature corrects for misalignment in the XYZ axes.
 * Take the following steps to get the bed skew in the XY plane:
 *  1. Print a test square (e.g., https://www.thingiverse.com/thing:2563185 )
 *  2. For XY_DIAG_AC measure the diagonal A to C
 *  3. For XY_DIAG_BD measure the diagonal B to D
 *  4. For XY_SIDE_AD measure the edge A to D
 *
 * Marlin automatically computes skew factors from these measurements.
 * Skew factors may also be computed and set manually:
 *
 *  - Compute AB     : SQRT(2*AC*AC+2*BD*BD-4*AD*AD)/2
 *  - XY_SKEW_FACTOR : TAN(PI/2-ACOS((AC*AC-AB*AB-AD*AD)/(2*AB*AD)))
 *
 * If desired, follow the same procedure for XZ and YZ.
 * Use these diagrams for reference:
 *
 *    Y                     Z                     Z
 *    ^     B-------C       ^     B-------C       ^     B-------C
 *    |    /       /        |    /       /        |    /       /
 *    |   /       /         |   /       /         |   /       /
 *    |  A-------D          |  A-------D          |  A-------D
 *    +-------------->X     +-------------->X     +-------------->Y
 *     XY_SKEW_FACTOR        XZ_SKEW_FACTOR        YZ_SKEW_FACTOR
 */
//#define SKEW_CORRECTION
#if ENABLED(SKEW_CORRECTION)
  // Input all length measurements here:
  #define XY_DIAG_AC 282.8427124746
  #define XY_DIAG_BD 282.8427124746
  #define XY_SIDE_AD 200
  // Or, set the default skew factors directly here
```

```
  // to override the above measurements:
  #define XY_SKEW_FACTOR 0.0
  //#define SKEW_CORRECTION_FOR_Z
  #if ENABLED(SKEW_CORRECTION_FOR_Z)
    #define XZ_DIAG_AC 282.8427124746
    #define XZ_DIAG_BD 282.8427124746
    #define YZ_DIAG_AC 282.8427124746
    #define YZ_DIAG_BD 282.8427124746
    #define YZ_SIDE_AD 200
    #define XZ_SKEW_FACTOR 0.0
    #define YZ_SKEW_FACTOR 0.0
  #endif
  // Enable this option for M852 to set skew at runtime
  //#define SKEW_CORRECTION_GCODE
#endif

//===========================================================================
//============================= Additional Features ===========================
//===========================================================================
// @section extras
/**
 * EEPROM
 *
 * Persistent storage to preserve configurable settings across reboots.
 *
 *   M500 - Store settings to EEPROM.
 *   M501 - Read settings from EEPROM. (i.e., Throw away unsaved changes)
 *   M502 - Revert settings to "factory" defaults. (Follow with M500 to init the EEPROM.)
 */
#define EEPROM_SETTINGS     // Persistent storage with M500 and M501
#define DISABLE_M503        // Saves ~2700 bytes of PROGMEM. Disable for release!
#define EEPROM_CHITCHAT       // Give feedback on EEPROM commands. Disable to save PROGMEM.
#define EEPROM_BOOT_SILENT    // Keep M503 quiet and only give errors during first load
#if ENABLED(EEPROM_SETTINGS)
  #define EEPROM_AUTO_INIT  // Init EEPROM automatically on any errors.
#endif
//
// Host Keepalive
//
// When enabled Marlin will send a busy status message to the host
// every couple of seconds when it can't accept commands.
//
#define HOST_KEEPALIVE_FEATURE       // Disable this if your host doesn't like keepalive messages
#define DEFAULT_KEEPALIVE_INTERVAL 2  // Number of seconds between "busy" messages. Set with M113.
#define BUSY_WHILE_HEATING           // Some hosts require "busy" messages even during heating
//
// G20/G21 Inch mode support
//
//#define INCH_MODE_SUPPORT
//
// M149 Set temperature units support
//
//#define TEMPERATURE_UNITS_SUPPORT
// @section temperature
//
// Preheat Constants - Up to 5 are supported without changes
//
#define PREHEAT_1_LABEL       "PLA"
#define PREHEAT_1_TEMP_HOTEND 180
#define PREHEAT_1_TEMP_BED     70
#define PREHEAT_1_TEMP_CHAMBER 35
```

```
#define PREHEAT_1_FAN_SPEED     0 // Value from 0 to 255
#define PREHEAT_2_LABEL       "ABS"
#define PREHEAT_2_TEMP_HOTEND 240
#define PREHEAT_2_TEMP_BED    110
#define PREHEAT_2_TEMP_CHAMBER 35
#define PREHEAT_2_FAN_SPEED     0 // Value from 0 to 255
/**

 * Nozzle Park
 *
 * Park the nozzle at the given XYZ position on idle or G27.
 *
 * The "P" parameter controls the action applied to the Z axis:
 *
 *    P0  (Default) If Z is below park Z raise the nozzle.
 *    P1  Raise the nozzle always to Z-park height.
 *    P2  Raise the nozzle by Z-park amount, limited to Z_MAX_POS.
 */
#define NOZZLE_PARK_FEATURE
#if ENABLED(NOZZLE_PARK_FEATURE)
  // Specify a park position as { X, Y, Z_raise }
  #define NOZZLE_PARK_POINT { (X_MIN_POS + 10), (Y_MAX_POS - 10), 20 }
  //#define NOZZLE_PARK_X_ONLY          // X move only is required to park
  //#define NOZZLE_PARK_Y_ONLY          // Y move only is required to park
  #define NOZZLE_PARK_Z_RAISE_MIN   2   // (mm) Always raise Z by at least this distance
  #define NOZZLE_PARK_XY_FEEDRATE 100   // (mm/s) X and Y axes feedrate (also used for delta Z axis)
  #define NOZZLE_PARK_Z_FEEDRATE    5   // (mm/s) Z axis feedrate (not used for delta printers)
#endif
/**
 * Clean Nozzle Feature – EXPERIMENTAL
 *
 * Adds the G12 command to perform a nozzle cleaning process.
 *
 * Parameters:
 *   P  Pattern
 *   S  Strokes / Repetitions
 *   T  Triangles (P1 only)
 *
 * Patterns:
 *   P0  Straight line (default). This process requires a sponge type material
 *       at a fixed bed location. "S" specifies strokes (i.e. back-forth motions)
 *       between the start / end points.
 *
 *   P1  Zig-zag pattern between (X0, Y0) and (X1, Y1), "T" specifies the
 *       number of zig-zag triangles to do. "S" defines the number of strokes.
 *       Zig-zags are done in whichever is the narrower dimension.
 *       For example, "G12 P1 S1 T3" will execute:
 *
 *           --
 *          |  (X0, Y1) |     /\         /\         /\     | (X1, Y1)
 *          |           |    /  \       /  \       /  \    |
 *        A |           |   /    \     /    \     /    \   |
 *          |           |  /      \   /      \   /      \  |
 *          |  (X0, Y0) | /        \ /        \ /        \ | (X1, Y0)
 *           --         +---------------------------------+
 *                        |________|_________|_________|
 *                            T1        T2        T3
 *
 *   P2  Circular pattern with middle at NOZZLE_CLEAN_CIRCLE_MIDDLE.
 *       "R" specifies the radius. "S" specifies the stroke count.
 *       Before starting, the nozzle moves to NOZZLE_CLEAN_START_POINT.
 *
```

```
 *   Caveats: The ending Z should be the same as starting Z.
 * Attention: EXPERIMENTAL. G-code arguments may change.
 */
//#define NOZZLE_CLEAN_FEATURE
#if ENABLED(NOZZLE_CLEAN_FEATURE)
  // Default number of pattern repetitions
  #define NOZZLE_CLEAN_STROKES  12
  // Default number of triangles
  #define NOZZLE_CLEAN_TRIANGLES  3
  // Specify positions for each tool as { { X, Y, Z }, { X, Y, Z } }
  // Dual hotend system may use
       { {  -20, (Y_BED_SIZE / 2), (Z_MIN_POS + 1) },  {  420, (Y_BED_SIZE / 2), (Z_MIN_POS + 1) }}
  #define NOZZLE_CLEAN_START_POINT { {  30, 30, (Z_MIN_POS + 1) } }
  #define NOZZLE_CLEAN_END_POINT   { { 100, 60, (Z_MIN_POS + 1) } }
  // Circular pattern radius
  #define NOZZLE_CLEAN_CIRCLE_RADIUS 6.5
  // Circular pattern circle fragments number
  #define NOZZLE_CLEAN_CIRCLE_FN 10
  // Middle point of circle
  #define NOZZLE_CLEAN_CIRCLE_MIDDLE NOZZLE_CLEAN_START_POINT
  // Move the nozzle to the initial position after cleaning
  #define NOZZLE_CLEAN_GOBACK
  // For a purge/clean station that's always at the gantry height (thus no Z move)
  //#define NOZZLE_CLEAN_NO_Z
  // For a purge/clean station mounted on the X axis
  //#define NOZZLE_CLEAN_NO_Y
  // Require a minimum hotend temperature for cleaning
  #define NOZZLE_CLEAN_MIN_TEMP 170
  //#define NOZZLE_CLEAN_HEATUP       // Heat up the nozzle instead of skipping wipe
  // Explicit wipe G-code script applies to a G12 with no arguments.
  //#define WIPE_SEQUENCE_COMMANDS "G1 X-17 Y25 Z10 F4000\nG1 Z1\nM114\nG1 X-17 Y25\nG1 X-17 Y95\nG1
X-17 Y25\nG1 X-17 Y95\nG1 X-17 Y25\nG1 X-17 Y95\nG1 X-17 Y25\nG1 X-17 Y95\nG1 X-17 Y25\nG1 X-17 Y95\nG1
X-17 Y25\nG1 X-17 Y95\nG1 Z15\nM400\nG0 X-10.0 Y-9.0"
#endif
/**
 * Print Job Timer
 *
 * Automatically start and stop the print job timer on M104/M109/M140/M190/M141/M191.
 * The print job timer will only be stopped if the bed/chamber target temp is
 * below BED_MINTEMP/CHAMBER_MINTEMP.
 *
 *   M104 (hotend, no wait)  - high temp = none,        low temp = stop timer
 *   M109 (hotend, wait)     - high temp = start timer, low temp = stop timer
 *   M140 (bed, no wait)     - high temp = none,        low temp = stop timer
 *   M190 (bed, wait)        - high temp = start timer, low temp = none
 *   M141 (chamber, no wait) - high temp = none,        low temp = stop timer
 *   M191 (chamber, wait)    - high temp = start timer, low temp = none
 *
 * For M104/M109, high temp is anything over EXTRUDE_MINTEMP / 2.
 * For M140/M190, high temp is anything over BED_MINTEMP.
 * For M141/M191, high temp is anything over CHAMBER_MINTEMP.
 *
 * The timer can also be controlled with the following commands:
 *
 *   M75 - Start the print job timer
 *   M76 - Pause the print job timer
 *   M77 - Stop the print job timer
 */
#define PRINTJOB_TIMER_AUTOSTART
/**
 * Print Counter
```

```
 *
 * Track statistical data such as:
 *
 *  - Total print jobs
 *  - Total successful print jobs
 *  - Total failed print jobs
 *  - Total time printing
 *
 * View the current statistics with M78.
 */
//#define PRINTCOUNTER
#if ENABLED(PRINTCOUNTER)
  #define PRINTCOUNTER_SAVE_INTERVAL 60 // (minutes) EEPROM save interval during print
#endif
/**
 * Password
 *
 * Set a numerical password for the printer which can be requested:
 *
 *  - When the printer boots up
 *  - Upon opening the 'Print from Media' Menu
 *  - When SD printing is completed or aborted
 *
 * The following G-codes can be used:
 *
 *  M510 - Lock Printer. Blocks all commands except M511.
 *  M511 - Unlock Printer.
 *  M512 - Set, Change and Remove Password.
 *
 * If you forget the password and get locked out you'll need to re-flash
 * the firmware with the feature disabled, reset EEPROM, and (optionally)
 * re-flash the firmware again with this feature enabled.
 */
//#define PASSWORD_FEATURE
#if ENABLED(PASSWORD_FEATURE)
  #define PASSWORD_LENGTH 4               // (#) Number of digits (1-9). 3 or 4 is recommended
  #define PASSWORD_ON_STARTUP
  #define PASSWORD_UNLOCK_GCODE   // Unlock with the M511 P<password> command. Disable to prevent
                                  // brute-force attack.
  #define PASSWORD_CHANGE_GCODE           // Change the password with M512 P<old> S<new>.
  //#define PASSWORD_ON_SD_PRINT_MENU       // This does not prevent gcodes from running
  //#define PASSWORD_AFTER_SD_PRINT_END
  //#define PASSWORD_AFTER_SD_PRINT_ABORT
  //#include "Configuration_Secure.h"       // External file with PASSWORD_DEFAULT_VALUE
#endif

//===========================================================================
//============================= LCD and SD support ==========================
//===========================================================================
// @section lcd
/**
 * LCD LANGUAGE
 *
 * Select the language to display on the LCD. These languages are available:
 *
 *   en, an, bg, ca, cz, da, de, el, el_CY, es, eu, fi, fr, gl, hr, hu, it,
 *   jp_kana, ko_KR, nl, pl, pt, pt_br, ro, ru, sk, sv, tr, uk, vi, zh_CN, zh_TW
 *
 * :{ 'en':'English', 'an':'Aragonese', 'bg':'Bulgarian', 'ca':'Catalan', 'cz':'Czech', 'da':'Danish', 'de':'German', 'el':'Greek
(Greece)', 'el_CY':'Greek (Cyprus)', 'es':'Spanish', 'eu':'Basque-Euskera', 'fi':'Finnish', 'fr':'French', 'gl':'Galician',
'hr':'Croatian', 'hu':'Hungarian', 'it':'Italian', 'jp_kana':'Japanese', 'ko_KR':'Korean (South Korea)', 'nl':'Dutch',
```

'pl':'Polish', 'pt':'Portuguese', 'pt_br':'Portuguese (Brazilian)', 'ro':'Romanian', 'ru':'Russian', 'sk':'Slovak', 'sv':'Swedish', 'tr':'Turkish', 'uk':'Ukrainian', 'vi':'Vietnamese', 'zh_CN':'Chinese (Simplified)', 'zh_TW':'Chinese (Traditional)' }
*/
#define LCD_LANGUAGE en // her kan der ændres til dansk ved indsættelse af "da" I stedet for "en"
/**
* LCD Character Set
*
* Note: This option is NOT applicable to Graphical Displays.
*
* All character-based LCDs provide ASCII plus one of these
* language extensions:
*
* - JAPANESE ... the most common
* - WESTERN ... with more accented characters
* - CYRILLIC ... for the Russian language
*
* To determine the language extension installed on your controller:
*
* - Compile and upload with LCD_LANGUAGE set to 'test'
* - Click the controller to view the LCD menu
* - The LCD will display Japanese, Western, or Cyrillic text
*
* See https://marlinfw.org/docs/development/lcd_language.html
*
* :['JAPANESE', 'WESTERN', 'CYRILLIC']
*/
#define DISPLAY_CHARSET_HD44780 JAPANESE
/**
* Info Screen Style (0:Classic, 1:Průša)
*
* :[0:'Classic', 1:'Průša']
*/
#define LCD_INFO_SCREEN_STYLE 0
/**
* SD CARD
*
* SD Card support is disabled by default. If your controller has an SD slot,
* you must uncomment the following option or it won't work.
*/
#define SDSUPPORT
/**
* SD CARD: ENABLE CRC
*
* Use CRC checks and retries on the SD communication.
*/
#define SD_CHECK_AND_RETRY
/**
* LCD Menu Items
*
* Disable all menus and only display the Status Screen, or
* just remove some extraneous menu items to recover space.
*/
//#define NO_LCD_MENUS
//#define SLIM_LCD_MENUS
//
// ENCODER SETTINGS
//
// This option overrides the default number of encoder pulses needed to
// produce one step. Should be increased for high-resolution encoders.
//

```
#define ENCODER_PULSES_PER_STEP 4
//
// Use this option to override the number of step signals required to
// move between next/prev menu items.
//
#define ENCODER_STEPS_PER_MENU_ITEM 1
/**
 * Encoder Direction Options
 *
 * Test your encoder's behavior first with both options disabled.
 *
 *  Reversed Value Edit and Menu Nav? Enable REVERSE_ENCODER_DIRECTION.
 *  Reversed Menu Navigation only?    Enable REVERSE_MENU_DIRECTION.
 *  Reversed Value Editing only?      Enable BOTH options.
 */
//
// This option reverses the encoder direction everywhere.
//
//  Set this option if CLOCKWISE causes values to DECREASE
//
#define REVERSE_ENCODER_DIRECTION
//
// This option reverses the encoder direction for navigating LCD menus.
//
//  If CLOCKWISE normally moves DOWN this makes it go UP.
//  If CLOCKWISE normally moves UP this makes it go DOWN.
//
//#define REVERSE_MENU_DIRECTION
//
// This option reverses the encoder direction for Select Screen.
//
//  If CLOCKWISE normally moves LEFT this makes it go RIGHT.
//  If CLOCKWISE normally moves RIGHT this makes it go LEFT.
//
//#define REVERSE_SELECT_DIRECTION
//
// Individual Axis Homing
//
// Add individual axis homing items (Home X, Home Y, and Home Z) to the LCD menu.
//
#define INDIVIDUAL_AXIS_HOMING_MENU
#define INDIVIDUAL_AXIS_HOMING_SUBMENU
//
// SPEAKER/BUZZER
//
// If you have a speaker that can produce tones, enable it here.
// By default Marlin assumes you have a buzzer with a fixed frequency.
//
#define SPEAKER
//
// The duration and frequency for the UI feedback sound.
// Set these to 0 to disable audio feedback in the LCD menus.
//
// Note: Test audio output with the G-Code:
//  M300 S<frequency Hz> P<duration ms>
//
//#define LCD_FEEDBACK_FREQUENCY_DURATION_MS 2
//#define LCD_FEEDBACK_FREQUENCY_HZ 5000
```

```
//===========================================================================
//========================== LCD / Controller Selection =====================
//========================   (Character-based LCDs)   =======================
//===========================================================================
//
// RepRapDiscount Smart Controller.
// https://reprap.org/wiki/RepRapDiscount_Smart_Controller
//
// Note: Usually sold with a white PCB.
//
//#define REPRAP_DISCOUNT_SMART_CONTROLLER
//
// GT2560 (YHCB2004) LCD Display
//
// Requires Testato, Koepel softwarewire library and
// Andriy Golovnya's LiquidCrystal_AIP31068 library.
//
//#define YHCB2004
//
// Original RADDS LCD Display+Encoder+SDCardReader
// http://doku.radds.org/dokumentation/lcd-display/
//
//#define RADDS_DISPLAY
//
// ULTIMAKER Controller.
//
//#define ULTIMAKERCONTROLLER
//
// ULTIPANEL as seen on Thingiverse.
//
//#define ULTIPANEL
//
// PanelOne from T3P3 (via RAMPS 1.4 AUX2/AUX3)
// https://reprap.org/wiki/PanelOne
//
//#define PANEL_ONE
//
// GADGETS3D G3D LCD/SD Controller
// https://reprap.org/wiki/RAMPS_1.3/1.4_GADGETS3D_Shield_with_Panel
//
// Note: Usually sold with a blue PCB.
//
//#define G3D_PANEL
//
// RigidBot Panel V1.0
// http://www.inventapart.com/
//
//#define RIGIDBOT_PANEL
//
// Makeboard 3D Printer Parts 3D Printer Mini Display 1602 Mini Controller
// https://www.aliexpress.com/item/32765887917.html
//
//#define MAKEBOARD_MINI_2_LINE_DISPLAY_1602
//
// ANET and Tronxy 20x4 Controller
//
//#define ZONESTAR_LCD           // Requires ADC_KEYPAD_PIN to be assigned to an analog pin.
                                 // This LCD is known to be susceptible to electrical interference
                                 // which scrambles the display.  Pressing any button clears it up.
                                 // This is a LCD2004 display with 5 analog buttons.
//
```

```
// Generic 16x2, 16x4, 20x2, or 20x4 character-based LCD.
//
//#define ULTRA_LCD
//===========================================================================
//========================= LCD / Controller Selection =========================
//=====================   (I2C and Shift-Register LCDs)   =====================
//===========================================================================
//
// CONTROLLER TYPE: I2C
//
// Note: These controllers require the installation of Arduino's LiquidCrystal_I2C
// library. For more info: https://github.com/kiyoshigawa/LiquidCrystal_I2C
//
//
// Elefu RA Board Control Panel
// http://www.elefu.com/index.php?route=product/product&product_id=53
//
//#define RA_CONTROL_PANEL
//
// Sainsmart (YwRobot) LCD Displays
//
// These require F.Malpartida's LiquidCrystal_I2C library
// https://bitbucket.org/fmalpartida/new-liquidcrystal/wiki/Home
//
//#define LCD_SAINSMART_I2C_1602
//#define LCD_SAINSMART_I2C_2004
//
// Generic LCM1602 LCD adapter
//
//#define LCM1602
//
// PANELOLU2 LCD with status LEDs,
// separate encoder and click inputs.
//
// Note: This controller requires Arduino's LiquidTWI2 library v1.2.3 or later.
// For more info: https://github.com/lincomatic/LiquidTWI2
//
// Note: The PANELOLU2 encoder click input can either be directly connected to
// a pin (if BTN_ENC defined to != -1) or read through I2C (when BTN_ENC == -1).
//
//#define LCD_I2C_PANELOLU2
//
// Panucatt VIKI LCD with status LEDs,
// integrated click & L/R/U/D buttons, separate encoder inputs.
//
//#define LCD_I2C_VIKI
//
// CONTROLLER TYPE: Shift register panels
//
//
// 2-wire Non-latching LCD SR from https://goo.gl/aJJ4sH
// LCD configuration: https://reprap.org/wiki/SAV_3D_LCD
//
//#define SAV_3DLCD
//
// 3-wire SR LCD with strobe using 74HC4094
// https://github.com/mikeshub/SailfishLCD
// Uses the code directly from Sailfish
//
//#define FF_INTERFACEBOARD
//
```

```
// TFT GLCD Panel with Marlin UI
// Panel connected to main board by SPI or I2C interface.
// See https://github.com/Serhiy-K/TFTGLCDAdapter
//
//#define TFTGLCD_PANEL_SPI
//#define TFTGLCD_PANEL_I2C
//==============================================================================
//=======================  LCD / Controller Selection  =========================
//=========================      (Graphical LCDs)      =========================
//==============================================================================
//
// CONTROLLER TYPE: Graphical 128x64 (DOGM)
//
// IMPORTANT: The U8glib library is required for Graphical Display!
//            https://github.com/olikraus/U8glib_Arduino
//
// NOTE: If the LCD is unresponsive you may need to reverse the plugs.
//
//
// RepRapDiscount FULL GRAPHIC Smart Controller
// https://reprap.org/wiki/RepRapDiscount_Full_Graphic_Smart_Controller
//
#define REPRAP_DISCOUNT_FULL_GRAPHIC_SMART_CONTROLLER
//
// K.3D Full Graphic Smart Controller
//
//#define K3D_FULL_GRAPHIC_SMART_CONTROLLER
//
// ReprapWorld Graphical LCD
// https://reprapworld.com/?products_details&products_id/1218
//
//#define REPRAPWORLD_GRAPHICAL_LCD
//
// Activate one of these if you have a Panucatt Devices
// Viki 2.0 or mini Viki with Graphic LCD
// https://www.panucatt.com
//
//#define VIKI2


//#define miniVIKI
//
// MakerLab Mini Panel with graphic
// controller and SD support - https://reprap.org/wiki/Mini_panel
//
//#define MINIPANEL
//
// MaKr3d Makr-Panel with graphic controller and SD support.
// https://reprap.org/wiki/MaKr3d_MaKrPanel
//
//#define MAKRPANEL
//
// Adafruit ST7565 Full Graphic Controller.
// https://github.com/eboston/Adafruit-ST7565-Full-Graphic-Controller/
//
//#define ELB_FULL_GRAPHIC_CONTROLLER
//
// BQ LCD Smart Controller shipped by
// default with the BQ Hephestos 2 and Witbox 2.
//
//#define BQ_LCD_SMART_CONTROLLER
```

```
//
// Cartesio UI
// http://mauk.cc/webshop/cartesio-shop/electronics/user-interface
//
//#define CARTESIO_UI
//
// LCD for Melzi Card with Graphical LCD
//
//#define LCD_FOR_MELZI
//
// Original Ulticontroller from Ultimaker 2 printer with SSD1309 I2C display and encoder
// https://github.com/Ultimaker/Ultimaker2/tree/master/1249_Ulticontroller_Board_(x1)
//
//#define ULTI_CONTROLLER
//
// MKS MINI12864 with graphic controller and SD support
// https://reprap.org/wiki/MKS_MINI_12864
//
//#define MKS_MINI_12864
//
// MKS MINI12864 V3 is an alias for FYSETC_MINI_12864_2_1. Type A/B. NeoPixel RGB Backlight.
//
//#define MKS_MINI_12864_V3
//
// MKS LCD12864A/B with graphic controller and SD support. Follows MKS_MINI_12864 pinout.
// https://www.aliexpress.com/item/33018110072.html
//
//#define MKS_LCD12864A
//#define MKS_LCD12864B
//
// FYSETC variant of the MINI12864 graphic controller with SD support
// https://wiki.fysetc.com/Mini12864_Panel/
//
//#define FYSETC_MINI_12864_X_X    // Type C/D/E/F. No tunable RGB Backlight by default
//#define FYSETC_MINI_12864_1_2    // Type C/D/E/F. Simple RGB Backlight (always on)
//#define FYSETC_MINI_12864_2_0    // Type A/B. Discreet RGB Backlight
//#define FYSETC_MINI_12864_2_1    // Type A/B. NeoPixel RGB Backlight
//#define FYSETC_GENERIC_12864_1_1 // Larger display with basic ON/OFF backlight.
//
// Factory display for Creality CR-10
// https://www.aliexpress.com/item/32833148327.html
//
// This is RAMPS-compatible using a single 10-pin connector.
// (For CR-10 owners who want to replace the Melzi Creality board but retain the display)
//
//#define CR10_STOCKDISPLAY
//
// Ender-2 OEM display, a variant of the MKS_MINI_12864
//
//#define ENDER2_STOCKDISPLAY
//
// ANET and Tronxy Graphical Controller
//
// Anet 128x64 full graphics lcd with rotary encoder as used on Anet A6
// A clone of the RepRapDiscount full graphics display but with
// different pins/wiring (see pins_ANET_10.h). Enable one of these.
//
//#define ANET_FULL_GRAPHICS_LCD
//#define ANET_FULL_GRAPHICS_LCD_ALT_WIRING
//
// AZSMZ 12864 LCD with SD
```

```
// https://www.aliexpress.com/item/32837222770.html
//
//#define AZSMZ_12864
//
// Silvergate GLCD controller
// https://github.com/android444/Silvergate
//
//#define SILVER_GATE_GLCD_CONTROLLER
//==============================================================================
//============================== OLED Displays ==================================
//==============================================================================
//
// SSD1306 OLED full graphics generic display
//
//#define U8GLIB_SSD1306
//
// SAV OLEd LCD module support using either SSD1306 or SH1106 based LCD modules
//
//#define SAV_3DGLCD
#if ENABLED(SAV_3DGLCD)
  #define U8GLIB_SSD1306
  //#define U8GLIB_SH1106
#endif
//
// TinyBoy2 128x64 OLED / Encoder Panel
//
//#define OLED_PANEL_TINYBOY2
//
// MKS OLED 1.3" 128×64 Full Graphics Controller
// https://reprap.org/wiki/MKS_12864OLED
//
// Tiny, but very sharp OLED display
//
//#define MKS_12864OLED          // Uses the SH1106 controller (default)
//#define MKS_12864OLED_SSD1306  // Uses the SSD1306 controller
//
// Zonestar OLED 128×64 Full Graphics Controller
//
//#define ZONESTAR_12864LCD           // Graphical (DOGM) with ST7920 controller
//#define ZONESTAR_12864OLED          // 1.3" OLED with SH1106 controller (default)
//#define ZONESTAR_12864OLED_SSD1306  // 0.96" OLED with SSD1306 controller
//
// Einstart S OLED SSD1306
//
//#define U8GLIB_SH1106_EINSTART
//
// Overlord OLED display/controller with i2c buzzer and LEDs
//
//#define OVERLORD_OLED
//
// FYSETC OLED 2.42" 128×64 Full Graphics Controller with WS2812 RGB
// Where to find : https://www.aliexpress.com/item/4000345255731.html
//#define FYSETC_242_OLED_12864   // Uses the SSD1309 controller
//
// K.3D SSD1309 OLED 2.42" 128×64 Full Graphics Controller
//
//#define K3D_242_OLED_CONTROLLER   // Software SPI
//==============================================================================
//========================== Extensible UI Displays ============================
//==============================================================================
//
```

```
// DGUS Touch Display with DWIN OS. (Choose one.)
// ORIGIN : https://www.aliexpress.com/item/32993409517.html
// FYSETC : https://www.aliexpress.com/item/32961471929.html
// MKS    : https://www.aliexpress.com/item/1005002008179262.html
//
// Flash display with DGUS Displays for Marlin:
//  - Format the SD card to FAT32 with an allocation size of 4kb.
//  - Download files as specified for your type of display.
//  - Plug the microSD card into the back of the display.
//  - Boot the display and wait for the update to complete.
//
// ORIGIN (Marlin DWIN_SET)
//  - Download https://github.com/coldtobi/Marlin_DGUS_Resources
//  - Copy the downloaded DWIN_SET folder to the SD card.
//
// FYSETC (Supplier default)
//  - Download https://github.com/FYSETC/FYSTLCD-2.0
//  - Copy the downloaded SCREEN folder to the SD card.
//
// HIPRECY (Supplier default)
//  - Download https://github.com/HiPrecy/Touch-Lcd-LEO
//  - Copy the downloaded DWIN_SET folder to the SD card.
//
// MKS (MKS-H43) (Supplier default)
//  - Download https://github.com/makerbase-mks/MKS-H43
//  - Copy the downloaded DWIN_SET folder to the SD card.
//
// RELOADED (T5UID1)
//  - Download https://github.com/Desuuuu/DGUS-reloaded/releases
//  - Copy the downloaded DWIN_SET folder to the SD card.
//
//#define DGUS_LCD_UI_ORIGIN
//#define DGUS_LCD_UI_FYSETC
//#define DGUS_LCD_UI_HIPRECY
//#define DGUS_LCD_UI_MKS
//#define DGUS_LCD_UI_RELOADED
#if ENABLED(DGUS_LCD_UI_MKS)
  #define USE_MKS_GREEN_UI
#endif
//
// Touch-screen LCD for Malyan M200/M300 printers
//
//#define MALYAN_LCD
#if ENABLED(MALYAN_LCD)
  #define LCD_SERIAL_PORT 1  // Default is 1 for Malyan M200
#endif
//
// Touch UI for FTDI EVE (FT800/FT810) displays
// See Configuration_adv.h for all configuration options.
//
//#define TOUCH_UI_FTDI_EVE
//
// Touch-screen LCD for Anycubic printers
//
//#define ANYCUBIC_LCD_I3MEGA
//#define ANYCUBIC_LCD_CHIRON
#if EITHER(ANYCUBIC_LCD_I3MEGA, ANYCUBIC_LCD_CHIRON)
  #define LCD_SERIAL_PORT 3  // Default is 3 for Anycubic
  //#define ANYCUBIC_LCD_DEBUG
#endif
//
```

```
// 320x240 Nextion 2.8" serial TFT Resistive Touch Screen NX3224T028
//
//#define NEXTION_TFT
#if ENABLED(NEXTION_TFT)
  #define LCD_SERIAL_PORT 1  // Default is 1 for Nextion
#endif
//
// Third-party or vendor-customized controller interfaces.
// Sources should be installed in 'src/lcd/extui'.
//
//#define EXTENSIBLE_UI
#if ENABLED(EXTENSIBLE_UI)
  //#define EXTUI_LOCAL_BEEPER // Enables use of local Beeper pin with external display
#endif
//==============================================================================
//============================== Graphical TFTs ================================
//==============================================================================
/**
 * Specific TFT Model Presets. Enable one of the following options
 * or enable TFT_GENERIC and set sub-options.
 */
//
// 480x320, 3.5", SPI Display From MKS
// Normally used in MKS Robin Nano V2
//
//#define MKS_TS35_V2_0
//
// 320x240, 2.4", FSMC Display From MKS
// Normally used in MKS Robin Nano V1.2
//
//#define MKS_ROBIN_TFT24
//
// 320x240, 2.8", FSMC Display From MKS
// Normally used in MKS Robin Nano V1.2
//
//#define MKS_ROBIN_TFT28
//
// 320x240, 3.2", FSMC Display From MKS
// Normally used in MKS Robin Nano V1.2
//
//#define MKS_ROBIN_TFT32
//
// 480x320, 3.5", FSMC Display From MKS

// Normally used in MKS Robin Nano V1.2
//
//#define MKS_ROBIN_TFT35
//
// 480x272, 4.3", FSMC Display From MKS
//
//#define MKS_ROBIN_TFT43
//
// 320x240, 3.2", FSMC Display From MKS
// Normally used in MKS Robin
//
//#define MKS_ROBIN_TFT_V1_1R
//
// 480x320, 3.5", FSMC Stock Display from TronxXY
//
//#define TFT_TRONXY_X5SA
//
```

```
// 480x320, 3.5", FSMC Stock Display from AnyCubic
//
//#define ANYCUBIC_TFT35
//
// 320x240, 2.8", FSMC Stock Display from Longer/Alfawise
//
//#define LONGER_LK_TFT28
//
// 320x240, 2.8", FSMC Stock Display from ET4
//
//#define ANET_ET4_TFT28
//
// 480x320, 3.5", FSMC Stock Display from ET5
//
//#define ANET_ET5_TFT35
//
// 1024x600, 7", RGB Stock Display from BIQU-BX
//
//#define BIQU_BX_TFT70
//
// Generic TFT with detailed options
//
//#define TFT_GENERIC
#if ENABLED(TFT_GENERIC)
  // :[ 'AUTO', 'ST7735', 'ST7789', 'ST7796', 'R61505', 'ILI9328', 'ILI9341', 'ILI9488' ]
  #define TFT_DRIVER AUTO
  // Interface. Enable one of the following options:
  //#define TFT_INTERFACE_FSMC
  //#define TFT_INTERFACE_SPI
  // TFT Resolution. Enable one of the following options:
  //#define TFT_RES_320x240
  //#define TFT_RES_480x272
  //#define TFT_RES_480x320
  //#define TFT_RES_1024x600
#endif
/**
 * TFT UI - User Interface Selection. Enable one of the following options:
 *
 *   TFT_CLASSIC_UI - Emulated DOGM - 128x64 Upscaled
 *   TFT_COLOR_UI   - Marlin Default Menus, Touch Friendly, using full TFT capabilities
 *   TFT_LVGL_UI    - A Modern UI using LVGL
 *
 *   For LVGL_UI also copy the 'assets' folder from the build directory to the
 *   root of your SD card, together with the compiled firmware.
 */
//#define TFT_CLASSIC_UI
//#define TFT_COLOR_UI
//#define TFT_LVGL_UI
#if ENABLED(TFT_LVGL_UI)
  //#define MKS_WIFI_MODULE  // MKS WiFi module
#endif
/**
 * TFT Rotation. Set to one of the following values:
 *
 *   TFT_ROTATE_90,  TFT_ROTATE_90_MIRROR_X,  TFT_ROTATE_90_MIRROR_Y,
 *   TFT_ROTATE_180, TFT_ROTATE_180_MIRROR_X, TFT_ROTATE_180_MIRROR_Y,
 *   TFT_ROTATE_270, TFT_ROTATE_270_MIRROR_X, TFT_ROTATE_270_MIRROR_Y,
 *   TFT_MIRROR_X, TFT_MIRROR_Y, TFT_NO_ROTATION
 */
//#define TFT_ROTATION TFT_NO_ROTATION
//===========================================================================
```

```
//============================ Other Controllers ============================
//===========================================================================
//
// Ender-3 v2 OEM display. A DWIN display with Rotary Encoder.
//
//#define DWIN_CREALITY_LCD
//
// Ender-3 v2 OEM display, enhanced.
//
//#define DWIN_CREALITY_LCD_ENHANCED
//
// Ender-3 v2 OEM display with enhancements by Jacob Myers
//
//#define DWIN_CREALITY_LCD_JYERSUI
//
// MarlinUI for Creality's DWIN display (and others)
//
//#define DWIN_MARLINUI_PORTRAIT
//#define DWIN_MARLINUI_LANDSCAPE
//
// Touch Screen Settings
//
//#define TOUCH_SCREEN
#if ENABLED(TOUCH_SCREEN)
  #define BUTTON_DELAY_EDIT  50 // (ms) Button repeat delay for edit screens
  #define BUTTON_DELAY_MENU 250 // (ms) Button repeat delay for menus
  //#define TOUCH_IDLE_SLEEP 300 // (secs) Turn off the TFT backlight if set (5mn)
  #define TOUCH_SCREEN_CALIBRATION
  //#define TOUCH_CALIBRATION_X 12316
  //#define TOUCH_CALIBRATION_Y -8981
  //#define TOUCH_OFFSET_X       -43
  //#define TOUCH_OFFSET_Y       257
  //#define TOUCH_ORIENTATION TOUCH_LANDSCAPE
  #if BOTH(TOUCH_SCREEN_CALIBRATION, EEPROM_SETTINGS)
    #define TOUCH_CALIBRATION_AUTO_SAVE // Auto save successful calibration values to EEPROM
  #endif
  #if ENABLED(TFT_COLOR_UI)
    //#define SINGLE_TOUCH_NAVIGATION
  #endif
#endif
//
// RepRapWorld REPRAPWORLD_KEYPAD v1.1
// https://reprapworld.com/products/electronics/ramps/keypad_v1_0_fully_assembled/
//

//#define REPRAPWORLD_KEYPAD
//#define REPRAPWORLD_KEYPAD_MOVE_STEP 10.0 // (mm) Distance to move per key-press
//===========================================================================
//============================== Extra Features ==============================
//===========================================================================
// @section extras
// Set number of user-controlled fans. Disable to use all board-defined fans.
// :[1,2,3,4,5,6,7,8]
//#define NUM_M106_FANS 1
// Increase the FAN PWM frequency. Removes the PWM noise but increases heating in the FET/Arduino
//#define FAST_PWM_FAN
// Use software PWM to drive the fan, as for the heaters. This uses a very low frequency
// which is not as annoying as with the hardware PWM. On the other hand, if this frequency
// is too low, you should also increment SOFT_PWM_SCALE.
//#define FAN_SOFT_PWM
// Incrementing this by 1 will double the software PWM frequency,
```

```
// affecting heaters, and the fan if FAN_SOFT_PWM is enabled.
// However, control resolution will be halved for each increment;
// at zero value, there are 128 effective control positions.
// :[0,1,2,3,4,5,6,7]
#define SOFT_PWM_SCALE 0
// If SOFT_PWM_SCALE is set to a value higher than 0, dithering can
// be used to mitigate the associated resolution loss. If enabled,
// some of the PWM cycles are stretched so on average the desired
// duty cycle is attained.
//#define SOFT_PWM_DITHER
// Temperature status LEDs that display the hotend and bed temperature.
// If all hotends, bed temperature, and target temperature are under 54C
// then the BLUE led is on. Otherwise the RED led is on. (1C hysteresis)
//#define TEMP_STAT_LEDS
// Support for the BariCUDA Paste Extruder
//#define BARICUDA
// Support for BlinkM/CyzRgb
//#define BLINKM
// Support for PCA9632 PWM LED driver
//#define PCA9632
// Support for PCA9533 PWM LED driver
//#define PCA9533
/**
** RGB LED / LED Strip Control
 *
 * Enable support for an RGB LED connected to 5V digital pins, or
 * an RGB Strip connected to MOSFETs controlled by digital pins.
 *
 * Adds the M150 command to set the LED (or LED strip) color.
 * If pins are PWM capable (e.g., 4, 5, 6, 11) then a range of
 * luminance values can be set from 0 to 255.
 * For NeoPixel LED an overall brightness parameter is also available.
 *
 * *** CAUTION ***
 *  LED Strips require a MOSFET Chip between PWM lines and LEDs,
 *  as the Arduino cannot handle the current the LEDs will require.
 *  Failure to follow this precaution can destroy your Arduino!
 *  NOTE: A separate 5V power supply is required! The NeoPixel LED needs
 *  more current than the Arduino 5V linear regulator can produce.
 * *** CAUTION ***
 *
 * LED Type. Enable only one of the following two options.
 */
//#define RGB_LED
//#define RGBW_LED
#if EITHER(RGB_LED, RGBW_LED)
  //#define RGB_LED_R_PIN 34
  //#define RGB_LED_G_PIN 43
  //#define RGB_LED_B_PIN 35
  //#define RGB_LED_W_PIN -1
#endif
// Support for Adafruit NeoPixel LED driver
//#define NEOPIXEL_LED
#if ENABLED(NEOPIXEL_LED)
  #define NEOPIXEL_TYPE   NEO_GRBW // NEO_GRBW / NEO_GRB - four/three channel driver type
                                   //(defined in Adafruit_NeoPixel.h)
  //#define NEOPIXEL_PIN     4     // LED driving pin
  //#define NEOPIXEL2_TYPE NEOPIXEL_TYPE
  //#define NEOPIXEL2_PIN    5
  #define NEOPIXEL_PIXELS 30       // Number of LEDs in the strip.
                                   // (Longest strip when NEOPIXEL2_SEPARATE is disabled.)
```

```
  #define NEOPIXEL_IS_SEQUENTIAL   // Sequential display for temperature change - LED by LED.
                                   // Disable to change all LEDs at once.
  #define NEOPIXEL_BRIGHTNESS 127  // Initial brightness (0-255)
  //#define NEOPIXEL_STARTUP_TEST  // Cycle through colors at startup
  // Support for second Adafruit NeoPixel LED driver controlled with M150 S1 ...
  //#define NEOPIXEL2_SEPARATE
  #if ENABLED(NEOPIXEL2_SEPARATE)
    #define NEOPIXEL2_PIXELS      15  // Number of LEDs in the second strip
    #define NEOPIXEL2_BRIGHTNESS 127  // Initial brightness (0-255)
    #define NEOPIXEL2_STARTUP_TEST    // Cycle through colors at startup
  #else
    //#define NEOPIXEL2_INSERIES      // Default behavior is NeoPixel 2 in parallel
  #endif
  // Use some of the NeoPixel LEDs for static (background) lighting
  //#define NEOPIXEL_BKGD_INDEX_FIRST  0            // Index of the first background LED
  //#define NEOPIXEL_BKGD_INDEX_LAST   5            // Index of the last background LED
  //#define NEOPIXEL_BKGD_COLOR { 255, 255, 255, 0 }  // R, G, B, W
  //#define NEOPIXEL_BKGD_ALWAYS_ON                 // Keep the backlight on when other NeoPixels are off
#endif

/**
 *
 * Printer Event LEDs
 *
 * During printing, the LEDs will reflect the printer status:
 *
 *  - Gradually change from blue to violet as the heated bed gets to target temp
 *  - Gradually change from violet to red as the hotend gets to temperature
 *  - Change to white to illuminate work surface
 *  - Change to green once print has finished
 *  - Turn off after the print has finished and the user has pushed a button
 */
#if ANY(BLINKM, RGB_LED, RGBW_LED, PCA9632, PCA9533, NEOPIXEL_LED)
  #define PRINTER_EVENT_LEDS
#endif
/**
 * Number of servos
 *
 * For some servo-related options NUM_SERVOS will be set automatically.
 * Set this manually if there are extra servos needing manual control.
 * Set to 0 to turn off servo support.
 */
//#define NUM_SERVOS 3 // Note: Servo index starts with 0 for M280-M282 commands
// (ms) Delay before the next move will start, to give the servo time to reach its target angle.
// 300ms is a good value but you can try less delay.

// If the servo can't reach the requested position, increase it.
#define SERVO_DELAY { 300 }
// Only power servos during movement, otherwise leave off to prevent jitter
//#define DEACTIVATE_SERVOS_AFTER_MOVE
// Edit servo angles with M281 and save to EEPROM with M500
//#define EDITABLE_SERVO_ANGLES
// Disable servo with M282 to reduce power consumption, noise, and heat when not in use
//#define SERVO_DETACH_GCODE
```