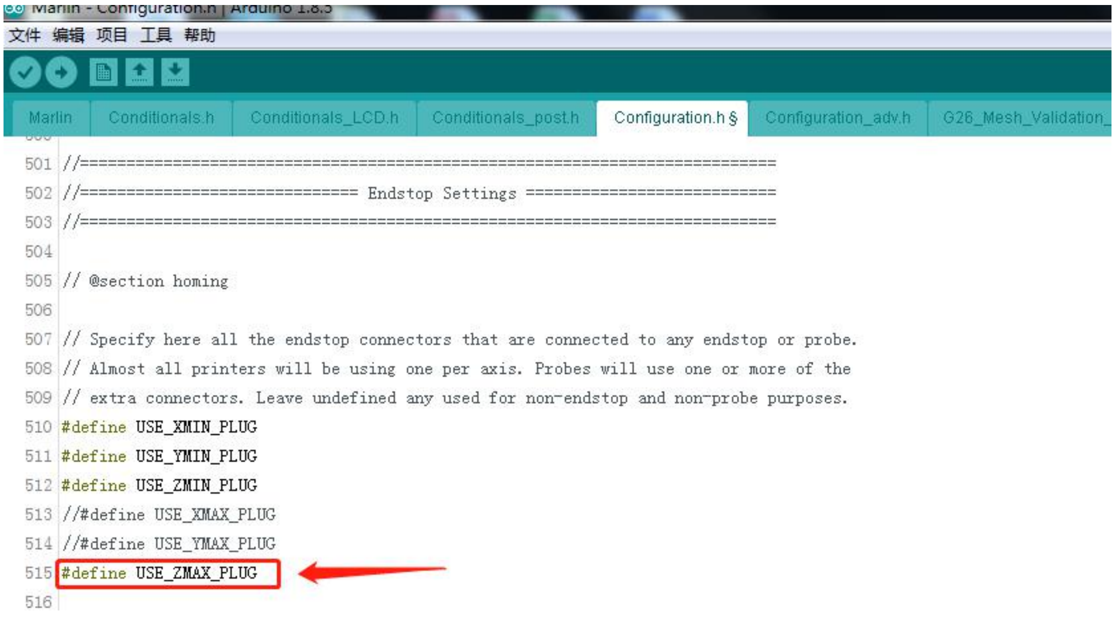
Z-PROBE auto-leveling

The ways to modify the Marlin firmware.

Step 1, Open configuration.h file, enable required enstop connector.

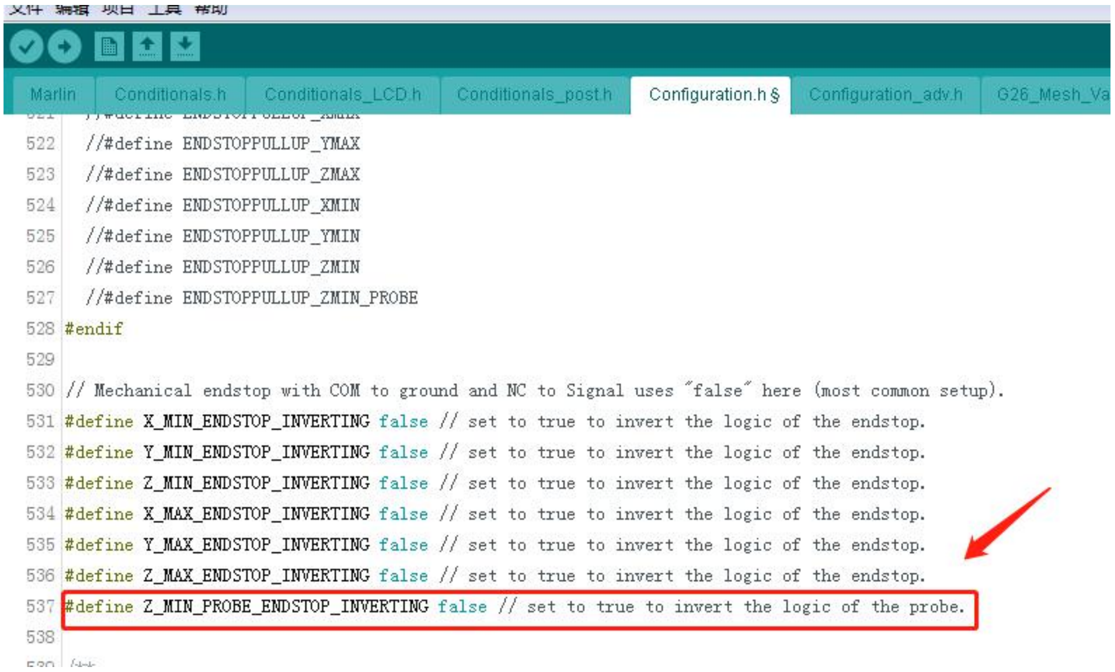
For example, if you want to use Z MAX as auto-leveling endstop connector, you can enable it on the Marlin Configuration.Please refer to the picture(Remove “//”).



```
//=============================================================================
//============================= 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_XMAX_PLUG
//#define USE_YMAX_PLUG
#define USE_ZMAX_PLUG

```

Step 2, Set true or false.

Mechanical endstop with COM to ground and NC to Signal uses "false" here (most common setup), inverting uses “true”.

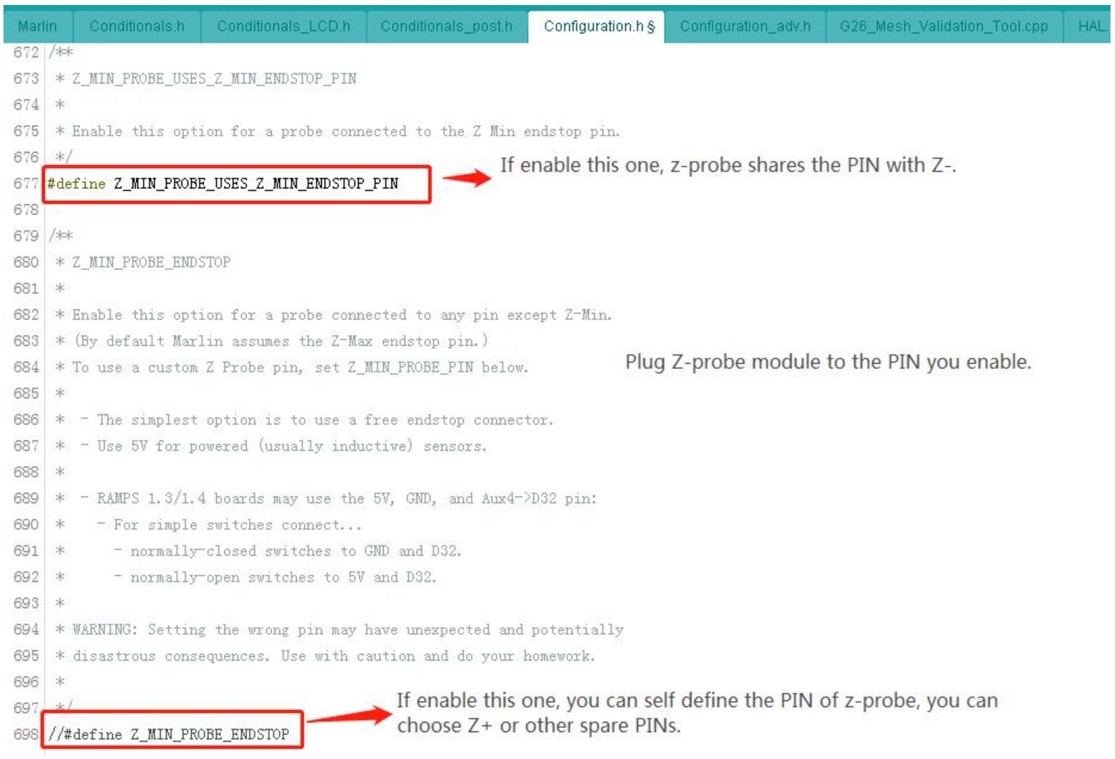


```
  //#define ENDSTOPPULLUP_YMAX
  //#define ENDSTOPPULLUP_ZMAX
  //#define ENDSTOPPULLUP_XMIN
  //#define ENDSTOPPULLUP_YMIN
  //#define ENDSTOPPULLUP_ZMIN
  //#define ENDSTOPPULLUP_ZMIN_PROBE
#endif

// Mechanical endstop with COM to ground and NC to Signal uses "false" here (most common setup).
#define X_MIN_ENDSTOP_INVERTING false // set to true to invert the logic of the endstop.
#define Y_MIN_ENDSTOP_INVERTING false // set to true to invert the logic of the endstop.
#define Z_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 Z_MIN_PROBE_ENDSTOP_INVERTING false // set to true to invert the logic of the probe.

```

Step 3, Take an example to describe it.

If you enable “#define Z_MIN_PROBE_USES_Z_MIN_ENDSTOP_PIN”, it means z-probe shares the

same PIN with Z-, that's to say, the z-probe module should connect to Z- endstop connector.
If you enable "#define Z_MIN_PROBE_ENDSTOP", you should define the PIN on "pin.ramps.h".



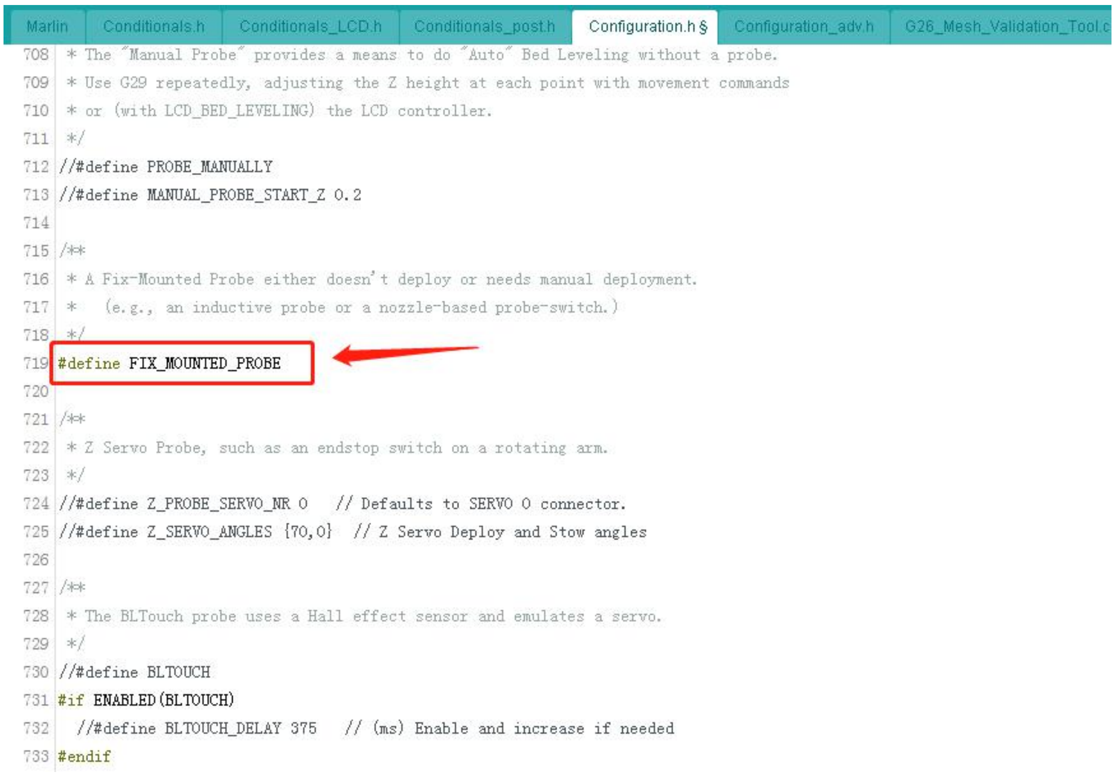
```
/**
 * Z_MIN_PROBE_USES_Z_MIN_ENDSTOP_PIN
 *
 * Enable this option for a probe connected to the Z Min endstop pin.
 */
#define Z_MIN_PROBE_USES_Z_MIN_ENDSTOP_PIN

/**
 * Z_MIN_PROBE_ENDSTOP
 *
 * Enable this option for a probe connected to any pin except Z-Min.
 * (By default Marlin assumes the Z-Max endstop pin.)
 * To use a custom Z Probe pin, set Z_MIN_PROBE_PIN below.
 *
 *  - 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.
 *
 * WARNING: Setting the wrong pin may have unexpected and potentially
 * disastrous consequences. Use with caution and do your homework.
 *
 */
//#define Z_MIN_PROBE_ENDSTOP
```

If enable this one, z-probe shares the PIN with Z-.

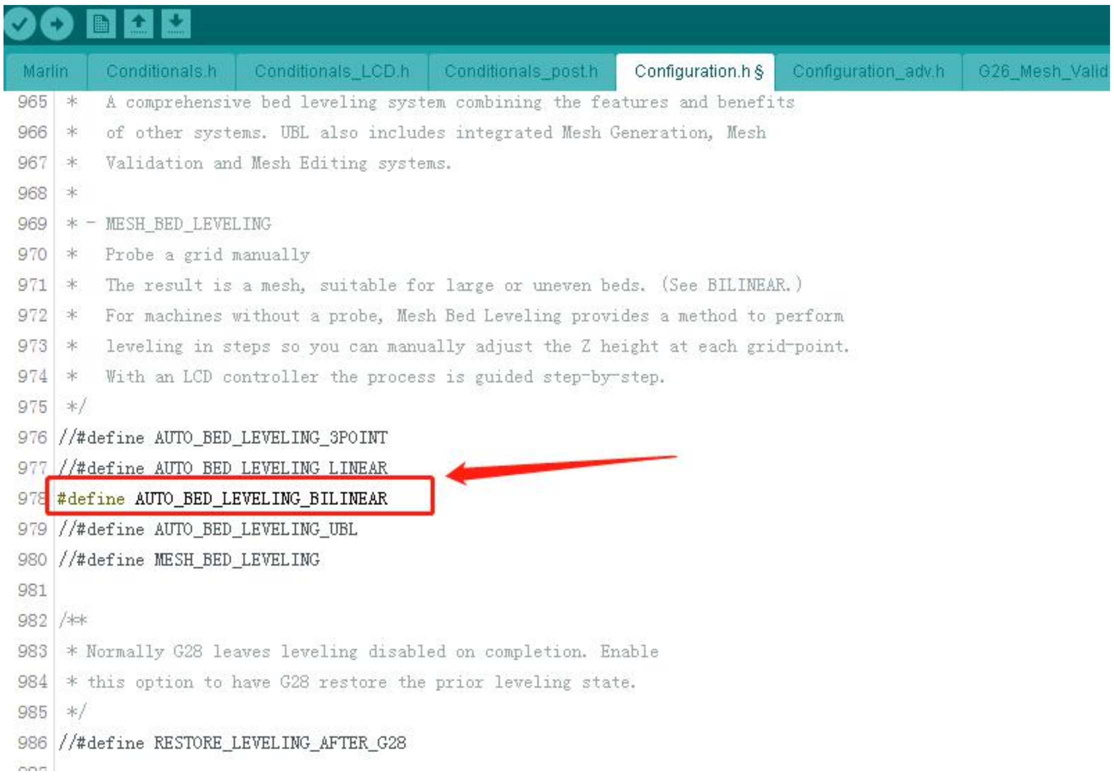
Plug Z-probe module to the PIN you enable.

If enable this one, you can self define the PIN of z-probe, you can choose Z+ or other spare PINs.

Step 4, Enable auto-leveling mode.

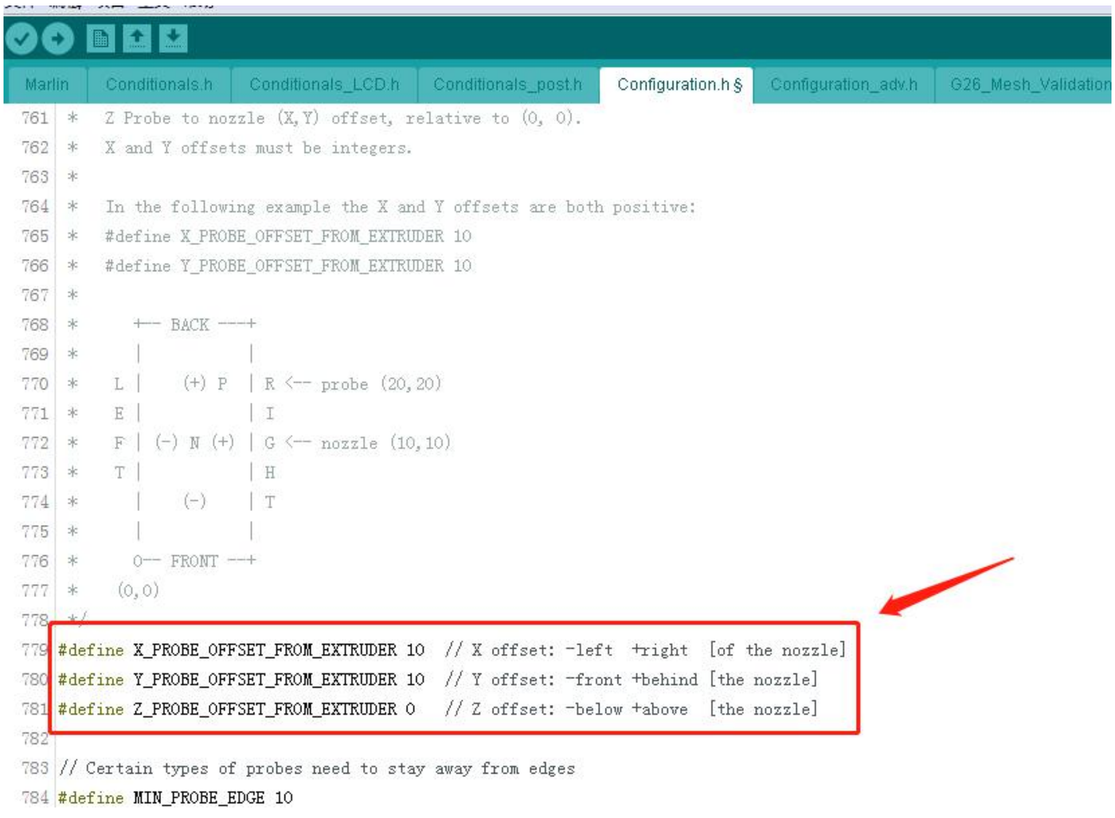


```
 * The "Manual Probe" provides a means to do "Auto" Bed Leveling without a probe.
 * Use G29 repeatedly, adjusting the Z height at each point with movement commands
 * or (with LCD_BED_LEVELING) the LCD controller.
 */
//#define PROBE_MANUALLY
//#define MANUAL_PROBE_START_Z 0.2

/**
 * 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

/**
 * 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
#if ENABLED(BLTOUCH)
  //#define BLTOUCH_DELAY 375   // (ms) Enable and increase if needed
#endif
```



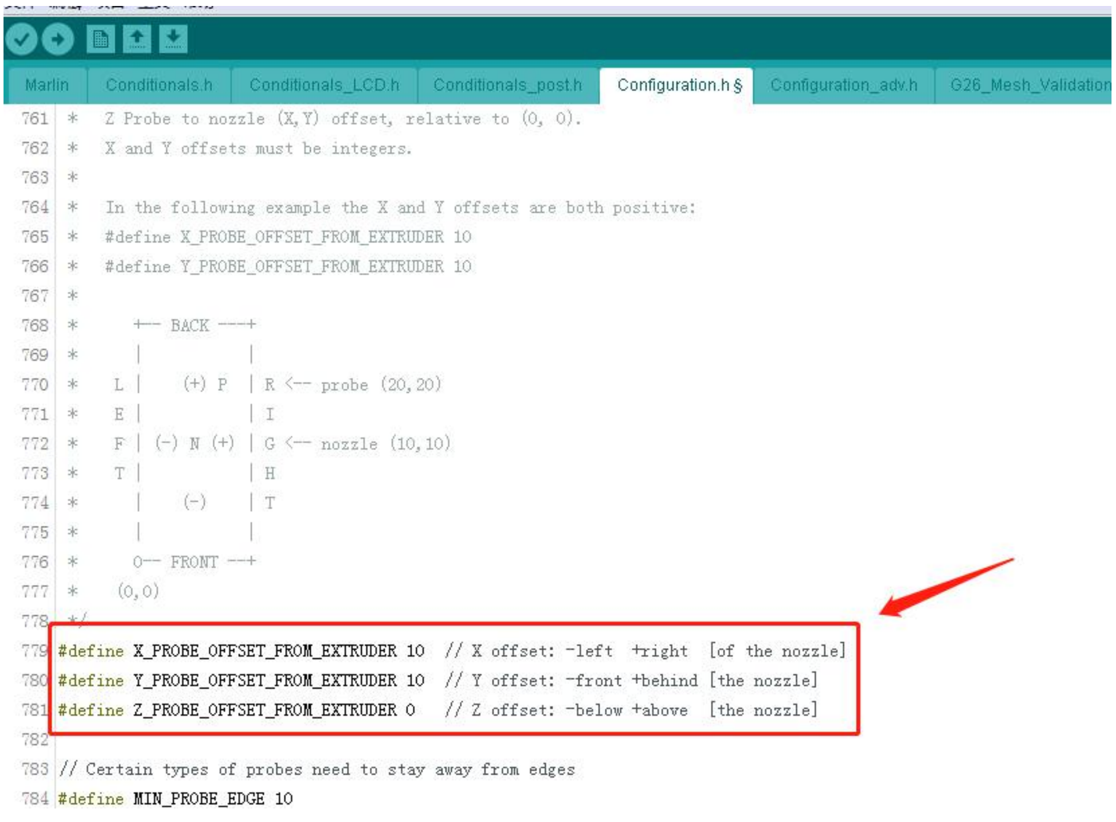
```
 *   A comprehensive bed leveling system combining the features and benefits
 *   of other systems. UBL also includes integrated Mesh Generation, Mesh
 *   Validation and Mesh Editing systems.
 *
 * - MESH_BED_LEVELING
 *   Probe a grid manually
 *   The result is a mesh, suitable for large or uneven 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
 * this option to have G28 restore the prior leveling state.
 */
//#define RESTORE_LEVELING_AFTER_G28
```

Step 5, Set the OFFSET of z-probe and extruders.



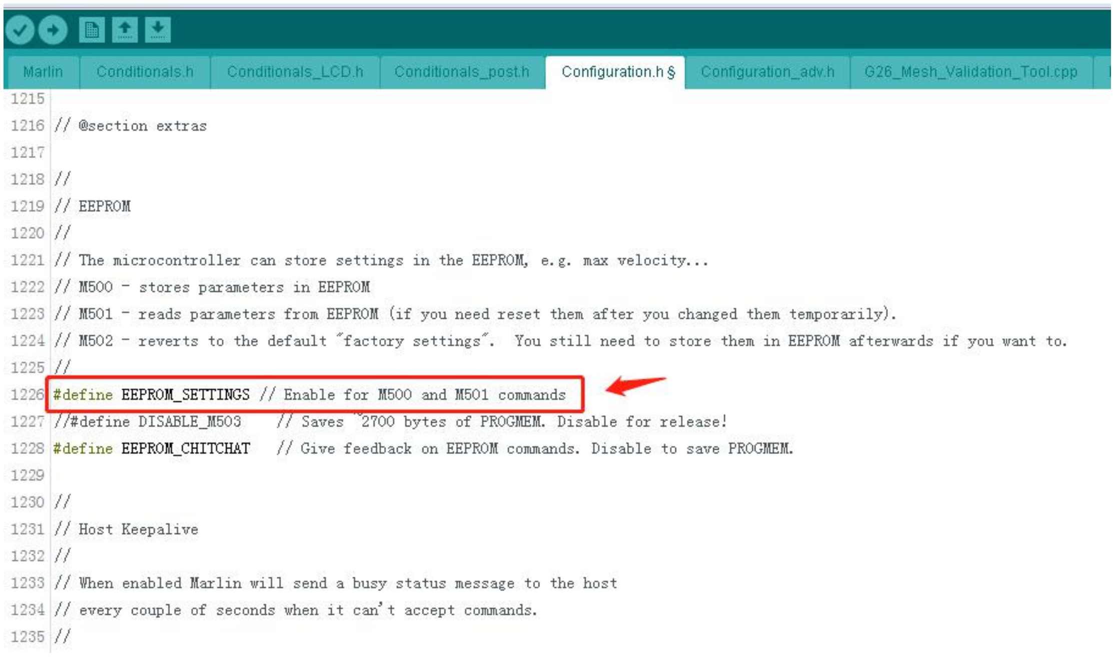
```
 *   Z Probe to nozzle (X,Y) offset, relative to (0, 0).
 *   X and Y offsets must be integers.
 *
 *   In the following example the X and Y offsets are both positive:
 *   #define X_PROBE_OFFSET_FROM_EXTRUDER 10
 *   #define Y_PROBE_OFFSET_FROM_EXTRUDER 10
 *
 *      +-- BACK ---+
 *      |           |
 *    L |    (+) P  | R <-- probe (20,20)
 *    E |           | I
 *    F | (-) N (+) | G <-- nozzle (10,10)
 *    T |           | H
 *      |    (-)    | T
 *      |           |
 *      O-- FRONT --+
 *    (0,0)
 */
#define X_PROBE_OFFSET_FROM_EXTRUDER 10  // X offset: -left  +right  [of the nozzle]
#define Y_PROBE_OFFSET_FROM_EXTRUDER 10  // Y offset: -front +behind [the nozzle]
#define Z_PROBE_OFFSET_FROM_EXTRUDER 0   // Z offset: -below +above  [the nozzle]

// Certain types of probes need to stay away from edges
#define MIN_PROBE_EDGE 10
```

Step 6, Set the points number and boundaries for probing.



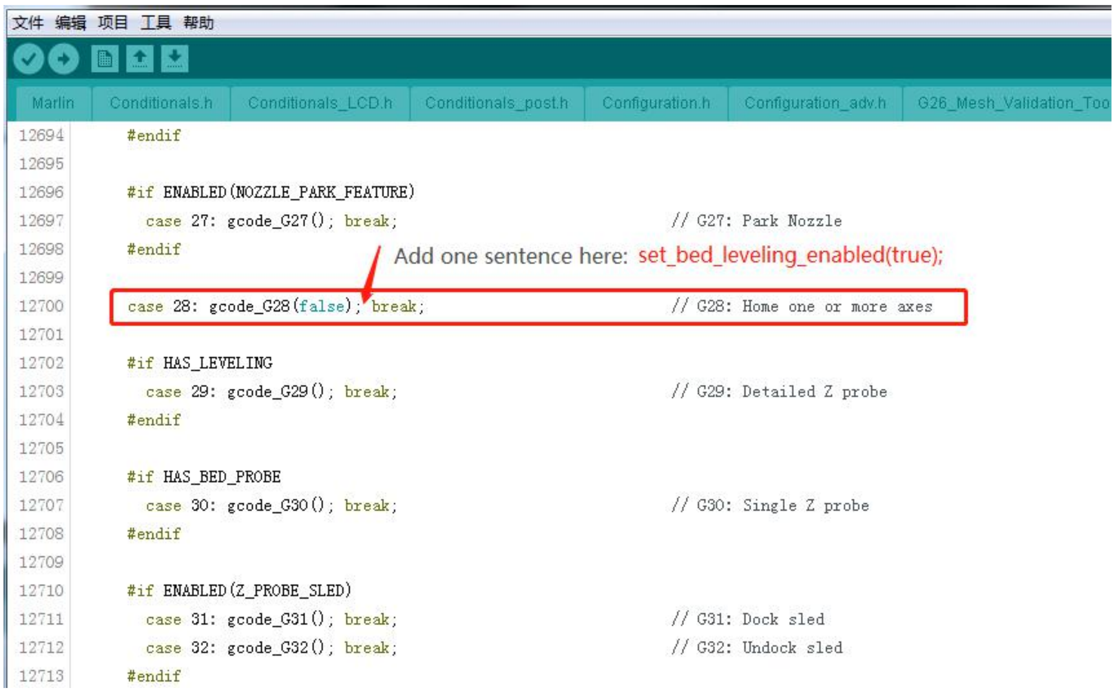
```
 *  Z Probe to nozzle (X,Y) offset, relative to (0, 0).
 *  X and Y offsets must be integers.
 *
 *  In the following example the X and Y offsets are both positive:
 *  #define X_PROBE_OFFSET_FROM_EXTRUDER 10
 *  #define Y_PROBE_OFFSET_FROM_EXTRUDER 10
 *
 *     +-- BACK ---+
 *     |           |
 *   L |    (+) P  | R <-- probe (20,20)
 *   E |           | I
 *   F | (-) N (+) | G <-- nozzle (10,10)
 *   T |           | H
 *     |    (-)    | T
 *     |           |
 *     O-- FRONT --+
 *   (0,0)
 */
#define X_PROBE_OFFSET_FROM_EXTRUDER 10  // X offset: -left  +right  [of the nozzle]
#define Y_PROBE_OFFSET_FROM_EXTRUDER 10  // Y offset: -front +behind [the nozzle]
#define Z_PROBE_OFFSET_FROM_EXTRUDER 0   // Z offset: -below +above  [the nozzle]

// Certain types of probes need to stay away from edges
#define MIN_PROBE_EDGE 10
```

Step 7, Enable "EEPROM_SETTING", enable M500 command.



```

// @section extras

//
// EEPROM
//
// The microcontroller can store settings in the EEPROM, e.g. max velocity...
// M500 - stores parameters in EEPROM
// M501 - reads parameters from EEPROM (if you need reset them after you changed them temporarily).
// M502 - reverts to the default "factory settings".  You still need to store them in EEPROM afterwards if you want to.
//
#define EEPROM_SETTINGS // Enable for M500 and M501 commands
//#define DISABLE_M503    // Saves ~2700 bytes of PROGMEM. Disable for release!
#define EEPROM_CHITCHAT   // Give feedback on EEPROM commands. Disable to save PROGMEM.

//
// Host Keepalive
//
// When enabled Marlin will send a busy status message to the host
// every couple of seconds when it can't accept commands.
//
```

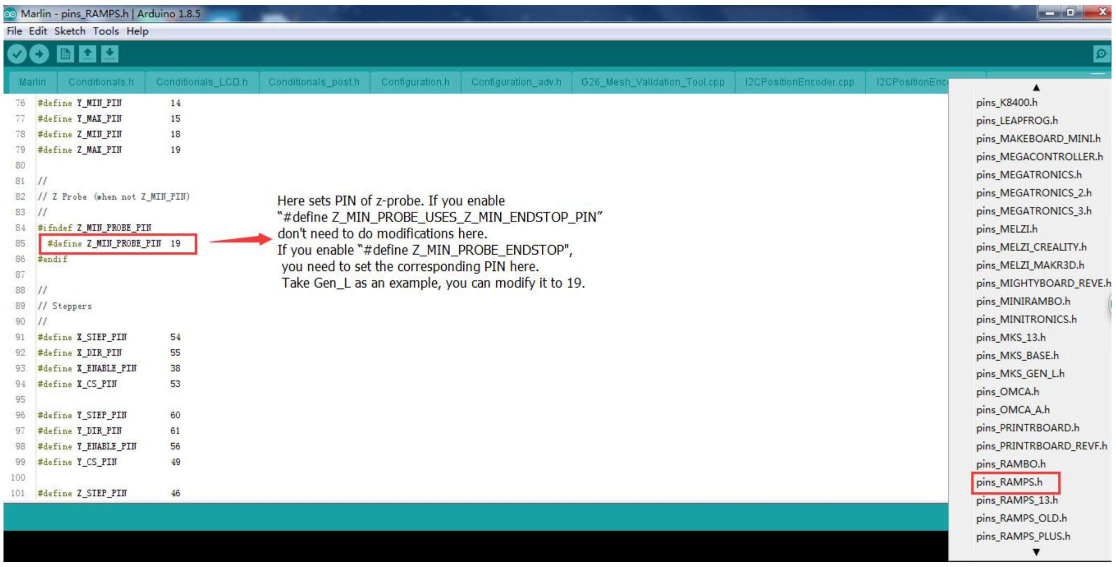
Step 8, open ***marlin_main.cpp*** file, find: "case 28: gcode_G28(false); break; // G28: Home one or more axes", and add this sentence: set_bed_leveling_enabled(true);



```
    #endif

    #if ENABLED(NOZZLE_PARK_FEATURE)
      case 27: gcode_G27(); break;                        // G27: Park Nozzle
    #endif

    case 28: gcode_G28(false); break;                     // G28: Home one or more axes

    #if HAS_LEVELING
      case 29: gcode_G29(); break;                        // G29: Detailed Z probe
    #endif

    #if HAS_BED_PROBE
      case 30: gcode_G30(); break;                        // G30: Single Z probe
    #endif

    #if ENABLED(Z_PROBE_SLED)
      case 31: gcode_G31(); break;                        // G31: Dock sled
      case 32: gcode_G32(); break;                        // G32: Undock sled
    #endif
```

Add one sentence here: set_bed_leveling_enabled(true);

Step 9, Open pin.ramps.h, as we said on step 3,
if you enable "#define Z_MIN_PROBE_ENDSTOP", you should define the PIN on "pin.ramps.h" as your actual need.



```
#define Y_MIN_PIN          14
#define Y_MAX_PIN          15
#define Z_MIN_PIN          18
#define Z_MAX_PIN          19

//
// Z Probe (when not Z_MIN_PIN)
//
#ifndef Z_MIN_PROBE_PIN
  #define Z_MIN_PROBE_PIN  19
#endif

//
// Steppers
//
#define X_STEP_PIN         54
#define X_DIR_PIN          55
#define X_ENABLE_PIN       38
#define X_CS_PIN           53

#define Y_STEP_PIN         60
#define Y_DIR_PIN          61
#define Y_ENABLE_PIN       56
#define Y_CS_PIN           49

#define Z_STEP_PIN         46
```

Here sets PIN of z-probe. If you enable "#define Z_MIN_PROBE_USES_Z_MIN_ENDSTOP_PIN" don't need to do modifications here.
If you enable "#define Z_MIN_PROBE_ENDSTOP", you need to set the corresponding PIN here.
Take Gen_L as an example, you can modify it to 19.

Completed above steps, update the firmware to the motherboard. Finished.