G-code

From RepRapWiki

English • العربية • български • català • česky • Deutsch • Ελληνικά • español • français • hrvatski • magyar • italiano • română • 日本語 • 한국어 • lietuvių • Nederlands • norsk bokmål • polski • português • русский • 中文(中国大陆) • 中文(台灣) • עברית •

This page tries to describe the flavour of G-codes that the RepRap firmwares use and how they work. The main target is additive fabrication using FFF/FDM processes. Codes for print head movements follow the NIST RS274NGC G-code standard (http://www.nist.gov/manuscript-publication-search.cfm?pub_id=823374) , so RepRap firmwares are quite usable for CNC milling and similar applications, too.

There are a few different ways to prepare GCode for a printer. One is to use a slicer like Slic3r, Skeinforge or Cura. These programs take a CAD model, slice it into layers, and output the GCode required for each layer. Slicers are the easiest way to go from a 3D model to a printed part, but the user sacrifices some flexibility when using them. Another option for GCode generation is to use a lower level library like mecode. Libraries like mecode give you precise control over the tool path, and thus are useful if you have a complex print that is not suitable for naive slicing. The final option is to just write the GCode yourself. This may be the best choice if you just need to run a few test lines while calibrating your printer.

As many different firmwares exist and their developers tend to implement new features without discussing strategies or looking what others did before them, a lot of different sub-flavours for the 3D-Printer specific codes developed over the years.

Contents

Introduction

A typical piece of GCode as sent to a RepRap machine might look like this:

```
N3 T0*57
N4 G92 E0*67
N5 G28*22
N6 G1 F1500.0*82
N7 G1 X2.0 Y2.0 F3000.0*85
N8 G1 X3.0 Y3.0*33
```

The meaning of all those symbols and numbers (and more) is explained below.

To find out which specific gcode/s are implemented in any given firmware, there are little tables attached to the command descriptions, like this one:

Support	FiveD	Teacup	Sprinter	Marlin	Repetier
	yes	automatic	yes	yes	experimental

Here means:

yes
: Fully supported.

experimental
: There is some support. Often it's required to check out a source code branch other than the default or to flip certain configuration switches.

automatic
: The firmware handles this feature automatically, so there's no need to send the command; you get the feature regardless. An example is power supply on/off (M80/M81) in Teacup firmware.

no
: The firmware doesn't support this feature.

For the technically minded, the end of line is marked by a <nl> and optionally a <cr>. So, Unix line endings work as well as Windows ones.

RepRap G Code Fields

This section explains the letter-preceded fields. The numbers in the fields are represented by nnn. Numbers can be integers, or can contain a decimal point, depending on context. For example an X coordinate can be integer (X175) or fractional (X17.62), whereas trying to select extruder number 2.76 would make no sense.

Letter	Meaning
Gnnn	Standard GCode command, such as move to a point
Mnnn	RepRap-defined command, such as turn on a cooling fan
Tnnn	Select tool nnn. In RepRap, tools are extruders
Snnn	Command parameter, such as the voltage to send to a motor
Pnnn	Command parameter, such as a time in milliseconds
Xnnn	An X coordinate, usually to move to
Ynnn	A Y coordinate, usually to move to
Znnn	A Z coordinate, usually to move to
Innn	Parameter - not currently used
Jnnn	Parameter - not currently used
Fnnn	Feedrate in mm per minute. (Speed of print head movement)
Rnnn	Parameter - used for temperatures
Qnnn	Parameter - not currently used
Ennn	Length of extrudate in mm. This is exactly like X, Y and Z, but for the length of filament to extrude. ~~It is common for newer stepper based systems to interpret ...~~ Better: Skeinforge 40 and up interprets this as the absolute length of input filament to consume, rather than the length of the extruded output.
Nnnn	Line number. Used to request repeat transmission in the case of communications errors.
*nnn	Checksum. Used to check for communications errors.

Comments

G Code comments begin at a semicolon, and end at the end of the line:

```
N3 T0*57 ;This is a comment
N4 G92 E0*67
; So is this
N5 G28*22
```

Will be ignored by RepRap, as will blank lines. But it's better to strip these out in the host computer before the lines are sent. This saves bandwidth.

Individual commands

Checking

N and *

Example: N123 [...G Code in here...] *71

These are the line number and the checksum. The RepRap firmware checks the checksum against a locally-computed value and, if they differ, requests a repeat transmission of the line of the given number.

You can leave both of these out - RepRap will still work, but it won't do checking. You have to have both or neither though.

The checksum "cs" for a GCode string "cmd" (including its line number) is computed by exor-ing the bytes in the string up to and not including the * character as follows:

```
int cs = 0;
for(i = 0; cmd[i] != '*' && cmd[i] != NULL; i++)
   cs = cs ^ cmd[i];
cs &= 0xff;  // Defensive programming...
```

and the value is appended as a decimal integer to the command after the * character.

The RepRap firmware expects line numbers to increase by 1 each line, and if that doesn't happen it is flagged as an error. But you can reset the count using M110 (see below).

Buffered G Commands

The RepRap firmware stores these commands in a ring buffer internally for execution. This means that there is no (appreciable) delay while a command is acknowledged and the next transmitted. In turn, this means that sequences of line segments can be plotted without a dwell between one and the next. As soon as one of these buffered commands is received it is acknowledged and stored locally. If the local buffer is full, then the acknowledgment is delayed until space for storage in the buffer is available. This is how flow control is achieved.

G0: Rapid move

Example: G0 X12

Support	FiveD	Teacup	Sprinter	Marlin	Repetier
	???	yes	yes	yes	yes

In this case move rapidly to X = 12 mm. In fact, the RepRap firmware uses exactly the same code for rapid as it uses for controlled moves (see G1 below), as - for the RepRap machine - this is just as efficient as not doing so. (The distinction comes from some old machine tools that used to move faster if the axes were not driven in a straight line. For them G0 allowed any movement in space to get to the destination as fast as possible.)

G1: Controlled move

Example: G1 X90.6 Y13.8 E22.4

Support	FiveD	Teacup	Sprinter	Marlin	Repetier
	yes	yes	yes	yes	yes

Go in a straight line from the current (X, Y) point to the point (90.6, 13.8), extruding material as the move happens from the current extruded length to a length of 22.4 mm.

RepRap does subtle things with feedrates. Thus:

```
G1 F1500
G1 X90.6 Y13.8 E22.4
```

Will set a feedrate of 1500 mm/minute, then do the move described above at that feedrate. But

```
G1 F1500
G1 X90.6 Y13.8 E22.4 F3000
```

Will set a feedrate of 1500 mm/minute, then do the move described above accelerating to a feedrate of 3000 mm/minute as it does so. The extrusion will accelerate along with the X, Y movement so everything stays synchronized.

RepRap thus treats feedrate as simply another variable (like X, Y, Z, and E) to be linearly interpolated. This gives complete control over accelerations and decelerations in a way that ensures that everything moves together and the right volume of material is extruded at all points.

Note: not every firmware implements this, e.g. the current Marlin will use the new feedrate from the beginning of the move and not change it.

The first example shows how to get a constant-speed movement. The second how to accelerate or decelerate. Thus

```
G1 F1500
G1 X90.6 Y13.8 E22.4 F3000
G1 X80 Y20 E36 F1500
```

Will do the first movement accelerating as before, and the second decelerating from 3000 mm/minute back to 1500 mm/minute.

To reverse the extruder by a given amount (for example to reduce its internal pressure while it does an in-air movement so that it doesn't dribble) simply use G1 to send an E value that is less than the currently extruded length.

Some implementations and RepRaps allow the sensing of endstops during moves to be switched on and off. Adding an S field allows this: G1 X300 S1 will move X to 300 checking for an endstop hit and stopping if it happens. G1 X300 S0 will do the same move with no checking. The default is no checking.

G2: Controlled Move Arc Clockwise

Example: G2 X90.6 Y13.8 I5 J10 E22.4

Support	FiveD	Teacup	Sprinter	Marlin	Repetier
	no	no	no	yes	yes

Go in a clockwise arc from the current (X, Y) point to the point (90.6, 13.8), while maintaining a constant distance from a center point at (5,10) from the current point, extruding material as the move happens from the current extruded length to a length of 22.4 mm.

G3: Controlled Move Arc Counter-Clockwise

Example: G3 X90.6 Y13.8 I5 J10 E22.4

Support	FiveD	Teacup	Sprinter	Marlin	Repetier
	no	no	no	yes	yes

Go in a counter-clockwise arc from the current (X, Y) point to the point (90.6, 13.8), while maintaining a constant distance from a center point at (5,10) from the current point, extruding material as the move happens from the current extruded length to a length of 22.4 mm.

G28: Move to Origin

Example: G28

Support	FiveD	Teacup	Sprinter	Marlin	Repetier
	yes	yes	yes	yes	yes

This causes the RepRap machine to move back to its X, Y and Z zero endstops, a process known as "homing". It does so accelerating, so as to get there fast. But when it arrives it backs off by 1 mm in each direction slowly, then moves back slowly to the stop. This ensures more accurate positioning.

If you add coordinates, then just the axes with coordinates specified will be zeroed. Thus

```
G28 X0 Y72.3
```

will zero the X and Y axes, but not Z. The actual coordinate values are ignored.

G29-G32: Bed probing

G29 Detailed Z-Probe

probes the bed at 3 points.

G30 Single Z Probe

In its simplest form probes bed at current XY location.

Some implementations allow more general behaviour: if a Pn field is specified the probed X, Y, and Z values are saved as point n on the bed for calculating the offset plane. Generally n is 0, 1, or 2. If X, or Y, or Z values are specified (e.g. G30 P1 X20 Y50 Z0.3) then those values are used instead of the machine's current coordinates. A silly Z value (less than -9999.0) causes the machine to probe at the current point to get Z, rather than using the given value. If an S field is specfied (e.g. G30 P1 Z0.3 S) the bed plane is computed for compensation and stored. The combination of these options allows for the machine to be moved to points using G1 commands, and then probe the bed, or for the user to position the nozzle interactively and use those coordinates. The user can also record those values and place them in a setup GCode file for automatic execution.

G31 Report Current Probe status

When used on its own this reports whether the Z probe is triggered, or gives the Z probe value in some units if the probe generates height values. If combined with a Z and P field (example: G31 P312 Z0.7) this will set the Z height to 0.7mm when the Z-probe value reaches 312 when a G28 Z0 (zero Z axis) command is sent. The machine will then move a further -0.7mm in Z to place itself at Z = 0. This allows non-contact measuring probes to approach but not touch the bed, and for the gap left to be allowed for. If the probe is a touch probe and generates a simple 0/1 off/on signal, then G31 Z0.7 will tell the RepRap machine that it is at a height of 0.7mm when the probe is triggered.

In Duet-dc42 firmware, separate G31 parameters may be defined for probe types 0, 1/2, and 3 (probe types 1 and 2 share the same set of parameters). To specify which probe you are setting parameters for, send a M558 command to select the probe type before the G31 command.

Duet-dc42 firmware supports additional parameters S (bed temperature in degC at which the specified Z parameter is correct, default is current bed temperature), and C (temperature coefficient of Z parameter in mm/degC, default zero). This is useful for ultrasonic and other probes that are affected by temperature.

G32 Probe Z and calculate Z plane

probes the bed at 3 or 4 pre-defined points (see M557) and updates transformation matrix for bed leveling compensation.

Unbuffered G commands

The following commands are not buffered. When one is received it is stored, but it is not acknowledged to the host until the buffer is exhausted and then the command has been executed. Thus the host will pause at one of these commands until it has been done. Short pauses between these commands and any that might follow them do not affect the performance of the machine.

Teacup Firmware buffers G20, G21, G90 and G91.

G4: Dwell

Example: G4 P200

In this case sit still doing nothing for 200 milliseconds. During delays the state of the machine (for example the temperatures of its extruders) will still be preserved and controlled.

On Marlin, the "S" parameter will wait for seconds, while the "P" parameter will wait for milliseconds. "G4 S2" and "G4 P2000" are equivalent.

G10: Tool Offset

Example: G10 P3 X17.8 Y-19.3 Z0.0 R140 S205

Support	RepRapPro	Teacup	Sprinter	Marlin	Repetier
	yes	no	no	(Retract?)	???

This sets the offset for tool (or in older implementations extrude head) 3 (from the P3) to the X and Y values specified. You can put a non-zero Z value in as well, but this is usually a bad idea unless the tools are loaded and unloaded by some sort of tool changer. When all the tools are in the machine at once they should all be set to the same Z height.

Remember that any parameter that you don't specify will automatically be set to the last value for that parameter. That usually means that you want explicitly to set Z0.0.

The R value is the standby temperature in ^{o}C that will be used for the tool, and the S value is its operating temperature. If you don't want the tool to be at a different temperature when not in use, set both values the same. See the T code (select tool) below. In tools with multiple heaters the temperatures for them all are specified thus: R100.0:90.0:20.0 S185.0:200.0:150.0 .

The NIST G-code standard (http://www.nist.gov/customcf/get_pdf.cfm?pub_id=823374) mentions an additional L parameter, which is ignored.

This command is subject to discussion.

Note that Marlin uses G10/G11 for executing a retraction/unretraction move.

G20: Set Units to Inches

Example: G20

Units from now on are in inches.

G21: Set Units to Millimeters

Example: G21

Units from now on are in millimeters. (This is the RepRap default.)

G90: Set to Absolute Positioning

Example: G90

All coordinates from now on are absolute relative to the origin of the machine. (This is the RepRap default.)

G91: Set to Relative Positioning

Example: G91

All coordinates from now on are relative to the last position.

G92: Set Position

Example: G92 X10 E90

Allows programming of absolute zero point, by reseting the current position to the values specified. This would set the machine's X coordinate to 10, and the extrude coordinate to 90. No physical motion will occur.

A G92 without coordinates will reset all axes to zero.

Unbuffered M and T commands

M0: Stop

Example: M0

The RepRap machine finishes any moves left in its buffer, then shuts down. All motors and heaters are turned off. It can be started again by pressing the reset button on the master microcontroller. See also M1, M112.

M1: Sleep

Example: M1

The RepRap machine finishes any moves left in its buffer, then shuts down. All motors and heaters are turned off. It can still be sent G and M codes, the first of which will wake it up again. See also M0, M112.

M3: Spindle On, Clockwise (CNC specific)

Example: M3 S4000

The spindle is turned on with a speed of 4000 RPM.

M4: Spindle On, Counter-Clockwise (CNC specific)

Example: M4 S4000

The spindle is turned on with a speed of 4000 RPM.

M5: Spindle Off (CNC specific)

Example: M5

The spindle is turned off.

M7: Mist Coolant On (CNC specific)

Example: M7

Mist coolant is turned on (if available)

M8: Flood Coolant On (CNC specific)

Example: M8

Flood coolant is turned on (if available)

M9: Coolant Off (CNC specific)

Example: M9

All coolant systems are turned off.

M10: Vacuum On (CNC specific)

Example: M10

Dust collection vacuum system turned on.

M11: Vacuum Off (CNC specific)

Example: M11

Dust collection vacuum system turned off.

M17: Enable/Power all stepper motors

Example: M17

M18: Disable all stepper motors

Example: M18

Disables stepper motors and allows axis to move 'freely.'

M20: List SD card

Example: M20

All files in the root folder of the SD card are listed to the serial port. This results in a line like:

ok Files: {SQUARE.G,SQCOM.G,}

The trailing comma is optional. Note that file names are returned in upper case, but - when sent to the M23 command (below) they must be in lower case. This seems to be a function of the SD software. Go figure...

M21: Initialize SD card

Example: M21

The SD card is initialized. If an SD card is loaded when the machine is switched on, this will happen by default. SD card must be initialized for the other SD functions to work.

M22: Release SD card

Example: M22

SD card is released and can be physically removed.

M23: Select SD file

Example: M23 filename.gco

The file specified as filename.gco (8.3 naming convention is supported) is selected ready for printing.

M24: Start/resume SD print

Example: M24

The machine prints from the file selected with the M23 command.

M25: Pause SD print

Example: M25

The machine pause printing at the current position within the file selected with the M23 command.

M26: Set SD position

Example: M26

Set SD position in bytes (M26 S12345).

M27: Report SD print status

Example: M27

Report SD print status.

M28: Begin write to SD card

Example: M28 filename.gco

File specified by filename.gco is created (or overwritten if it exists) on the SD card and all subsequent commands sent to the machine are written to that file.

M29: Stop writing to SD card

Example: M29 filename.gco

File opened by M28 command is closed, and all subsequent commands sent to the machine are executed as normal.

M30: Delete a file on the SD card

Example: M30 filename.gco

filename.gco is deleted.

M32: Select file and start SD print

(Can be used when printing from SD card)

Example: M32 filename.gco

tba available in marlin(14/6/2014)

M40: Eject

If your RepRap machine can eject the parts it has built off the bed, this command executes the eject cycle. This usually involves cooling the bed and then performing a sequence of movements that remove the printed parts from it. The X, Y and Z position of the machine at the end of this cycle are undefined (though they can be found out using the M114 command, q.v.).

See also M240 and M241 below.

M41: Loop

Example: M41

If the RepRap machine was building a file from its own memory such as a local SD card (as opposed to a file being transmitted to it from a host computer) this goes back to the beginning of the file and runs it again. So, for example, if your RepRap is capable of ejecting parts from its build bed then you can set it printing in a loop and it will run and run. Use with caution - the only things that will stop it are:

1. When you press the reset button,
2. When the build material runs out (if your RepRap is set up to detect this), and
3. When there's an error (such as a heater failure).

M42: Stop on material exhausted / Switch I/O pin

M42 in ???

Example: M42

If your RepRap can detect when its material runs out, this decides the behaviour when that happens. The X and Y axes are zeroed (but not Z), and then the machine shuts all motors and heaters off. You have to press reset to reactivate the machine. In other words, it parks itself and then executes an M0 command (q.v.).

M42 in Marlin/Sprinter

Example: M42 P7 S255

M42 switches a general purpose I/O pin. Use M42 Px Sy to set pin x to value y, when omitting Px the LEDPIN will be used.

M42 in Teacup

Not needed. General purpose devices are handled like a heater, see M104.

M43: Stand by on material exhausted

Example: M43

If your RepRap can detect when its material runs out, this decides the behaviour when that happens. The X and Y axes are zeroed (but not Z), and then the machine shuts all motors and heaters off except the heated bed, the temperature of which is maintained. The machine will still respond to G and M code commands in this state.

M80: ATX Power On

Example: M80

Turns on the ATX power supply from standby mode to fully operational mode. No-op on electronics without standby mode.

Note: some firmwares, like Teacup, handle power on/off automatically, so this is redundant there. Also, see RAMPS wiring for ATX on/off (http://forums.reprap.org/read.php?219,132664)

M81: ATX Power Off

Example: M81

Turns off the ATX power supply. Counterpart to M80.

M82: set extruder to absolute mode

Example: M82

makes the extruder interpret extrusion as absolute positions.

This is the default in repetier.

M83: set extruder to relative mode

Example: M83

makes the extruder interpret extrusion values as relative positions.

M84: Stop idle hold

Example: M84

Stop the idle hold on all axis and extruder. In some cases the idle hold causes annoying noises, which can be stopped by disabling the hold. Be aware that by disabling idle hold during printing, you will get quality issues. This is recommended only in between or after printjobs.

On Marlin, M84 can also be used to configure or disable the idle timeout. For example, "M84 S10" will idle the stepper motors after 10 seconds of inactivity. "M84 S0" will disable idle timeout; steppers will remain powered up regardless of activity.

M92: Set axis_steps_per_unit

Example: M92 X<newsteps> Sprinter and Marlin

Allows programming of steps per unit of axis till the electronics are reset for the specified axis. Very useful for calibration.

M98: Call Macro/Subprogram

Example: M98 Pmymacro.g

Runs the macro in the file mymacro.g. In conventional G Codes for CNC machines the P parameter normally refers to a line number in the program itself (P2000 would run the Macro starting at line O2000, say). For RepRap, which almost always has some sort of mass storage device inbuilt, it simply refers to the name of a GCode file that is executed by the G98 call. That GCode file does not need to end with an M99 (return) as the end-of-file automatically causes a return. It is usually a good idea to start a macro with an M120 (Push) instruction and to end it with an M121 (Pop) instruction, q.v. Macro calls cannot usually be nested or be recursive; i.e. you can't call a macro from a macro (though some implementations may allow this).

M99: Return from Macro/Subprogram

Example: M99

Returns from an M98 call.

M98: Get axis_hysteresis_mm

Deprecated - clashes with the G Code standard M98 above

Example: M98

Report the current hysteresis values in mm for all of the axis.

Proposed for Marlin

M99: Set axis_hysteresis_mm

Deprecated - clashes with the G Code standard M99 above

Example: M99 X<mm> Y<mm> Z<mm> E<mm>

Allows programming of axis hysteresis. Mechanical pulleys, gears and threads can have hysteresis when they change direction. That is, a certain number of steps occur before movement occurs. You can measure how many mm are lost to hysteresis and set their values with this command. Every time an axis changes direction, these extra mm will be added to compensate for the hysteresis.

Proposed for Marlin

M101 Turn extruder 1 on Forward / Undo Extruder Retraction

M101 in Teacup firmware

If a DC extruder is present, turn that on. Else, undo filament retraction, which means, make the extruder ready for extrusion. Complement to M103.

M101 in other firmwares

Deprecated. Regarding filament retraction, see M227, M228, M229.

M102 Turn extruder 1 on Reverse

Deprecated.

M103 Turn all extruders off / Extruder Retraction

M103 in Teacup firmware

If a DC extruder is present, turn that off. Else, retract the filament in the hope to prevent nozzle drooling. Complement to M101.

M103 in other firmwares

Deprecated. Regarding extruder retraction, see M227, M228, M229.

M104: Set Extruder Temperature

Example: M104 S190

Support	FiveD	Teacup	Sprinter	Marlin	Repetier
	yes	yes	yes	yes	yes

Set the temperature of the current extruder to 190^{o}C and return control to the host immediately (*i.e.* before that temperature has been reached by the extruder). See also M109.

This is deprecated because temperatures should be set using the G10 and T commands (q.v.).

Deprecation is subject to discussion. --Traumflug 11:33, 19 July 2012 (UTC)

M104 in Teacup Firmware

In Teacup Firmware, M104 can be additionally used to handle all devices using a temperature sensor. It supports the additional P parameter, which is a zero-based index into the list of sensors in config.h. For devices without a temp sensor, see M106.

Example: M104 P1 S100

Set the temperature of the device attached to the second temperature sensor to 100 °C.

M105: Get Extruder Temperature

Example: M105

Request the temperature of the current extruder and the build base in degrees Celsius. The temperatures are returned to the host computer. For example, the line sent to the host in response to this command looks like: `ok T:201 B:117`

Expansion/generalization of M105 to be considered as noted in Pronterface I/O Monitor

M106: Fan On

Example: M106 S127

Support	FiveD	Teacup	Sprinter	Marlin	Repetier
		yes	yes	yes	

Turn on the cooling fan at half speed.

Mandatory parameter 'S' declares the PWM value (0-255). M106 S0 turns the fan off. In some implementations the pwm is specified by a real fraction: M106 S0.7.

M106 in Teacup Firmware

Additionally to the above, Teacup Firmware uses M106 to control general devices. It supports the additional P parameter, which is an zero-based index into the list of heaters/devices in config.h.

Example: M106 P2 S255

Turn on device #3 at full speed/wattage.

Note: When turning on a temperature sensor equipped heater with M106 and M104 at the same time, temperature control will override the value given in M106 quickly.

M107: Fan Off

Deprecated. Use M106 S0 instead.

M108: Set Extruder Speed

Sets speed of extruder motor. (Deprecated in current firmware, see M113)

M109: Set Extruder Temperature and Wait

M109 in Teacup

Support	RepRapPro	Teacup	Sprinter	Marlin	Repetier
	(See G10)	not needed	see text	yes	???

Not needed. To mimic Marlin behaviour, use M104 followed by M116.

M109 in Marlin, Sprinter (ATmega port)

Set extruder heater temperature in degrees celsius and wait for this temperature to be achieved.

Example: M109 S185

M109 in Sprinter (4pi port)

Parameters: **S** (optional), set target temperature value. If not specified, waits for the temperature set by M104. **R** (optional), sets target temperature range maximum value.

Example: M109 S185 R240 //sets extruder temperature to 185 and waits for the temperature to be between 185 - 240.

If you have multiple extruders, use **T** or **P** parameter to specify which extruder you want to set/wait.

Another way to do this is to use G10.

M110: Set Current Line Number

Example: M110 N123

Set the current line number to 123. Thus the expected next line after this command will be 124.

Support	FiveD	Teacup	Sprinter	Marlin	Repetier
	???	not needed	???	???	???

M111: Set Debug Level

Example: M111 S6

Set the level of debugging information transmitted back to the host to level 6. The level is the OR of three bits:

```
#define DEBUG_ECHO (1<<0)
#define DEBUG_INFO (1<<1)
#define DEBUG_ERRORS (1<<2)
```

Thus 6 means send information and errors, but don't echo commands. (This is the RepRap default.)

For firmware that supports ethernet and web interfaces M111 S9 will turn web debug information on without changing any other debug settings, and M111 S8 will turn it off. Web debugging usually means that HTTP requests will be echoed to the USB interface, as will the responses.

Example: M253

Support	FiveD	Teacup	Sprinter	Marlin	Repetier
		Debug			

M112: Emergency Stop

Example: M112

Any moves in progress are immediately terminated, then RepRap shuts down. All motors and heaters are turned off. It can be started again by pressing the reset button on the master microcontroller. See also M0 and M1.

M113: Set Extruder PWM

Example: M113

Set the PWM for the currently-selected extruder. On its own this command sets RepRap to use the on-board potentiometer on the extruder controller board to set the PWM for the currently-selected extruder's stepper power. With an S field:

M113 S0.7

it causes the PWM to be set to the S value (70% in this instance). M113 S0 turns the extruder off, until an M113 command other than M113 S0 is sent.

M114: Get Current Position

Example: M114

This causes the RepRap machine to report its current X, Y, Z and E coordinates to the host.

For example, the machine returns a string such as:

```
ok C: X:0.00 Y:0.00 Z:0.00 E:0.00
```

In Marlin first 3 numbers is the position for the planner. The other positions are the positions from the stepper function. This helps for debugging a previous stepper function bug.

```
X:0.00 Y:0.00 RZ:0.00 LZ:0.00 Count X:0.00 Y:0.00 RZ:41.02 LZ:41.02
```

M115: Get Firmware Version and Capabilities

Example: M115

Request the Firmware Version and Capabilities of the current microcontroller The details are returned to the host computer as key:value pairs separated by spaces and terminated with a linefeed.

sample data from firmware:

```
ok PROTOCOL_VERSION:0.1 FIRMWARE_NAME:FiveD FIRMWARE_URL:http%3A//reprap.org MACHINE_TYPE:Mendel EXTRUDER_COUNT:1
```

This M115 code is inconsistently implemented, and should not be relied upon to exist, or output correctly in all cases. An initial implementation was committed to svn for the FiveD Reprap firmware on 11 Oct 2010. Work to more formally define protocol versions is currently (October 2010) being discussed. See M115_Keywords for one draft set of keywords and their meanings.

M116: Wait

Example: M116

Wait for *all* temperatures and other slowly-changing variables to arrive at their set values. See also M109.

Duet-dc42 firmware version 0.78c and later supports an optional P parameter, used to specify a tool number. If this parameter is present, then the system only waits for temperatures associated with that tool to arrive at their set values. This is useful during tool changes, to wait for the new tool to heat up without necessarily waiting for the old one to cool down fully.

M117: Get Zero Position

Example: M117

This causes the RepRap machine to report the X, Y, Z and E coordinates *in steps not mm* to the host that it found when it last hit the zero stops for those axes. That is to say, when you zero X, the *x* coordinate of the machine when it hits the X endstop is recorded. This value should be 0, of course. But if the machine has drifted (for example by dropping steps) then it won't be. This command allows you to measure and to diagnose such problems. (E is included for completeness. It doesn't normally have an endstop.)

M117 in Marlin: Display Message

Example: M117 Hello World

This causes the given message to be shown in the status line on an attached LCD. The above command will display Hello World.

M118: Negotiate Features

Example: M118 P42

This M-code is for future proofing. NO firmware or hostware supports this at the moment. It is used in conjunction with M115's FEATURES keyword.

See Protocol_Feature_Negotiation for more info.

M119: Get Endstop Status

Example: M119

Returns the current state of the configured X, Y, Z endstops. Takes into account any 'inverted endstop' settings, so one can confirm that the machine is interpreting the endstops correctly.

Support	**FiveD**	**Teacup**	**Sprinter**	**Marlin**	**Repetier**
		yes		yes	yes

M120: Push

Push the state of the RepRap machine onto a stack. Exactly what variables get pushed depends on the implementation (as does the depth of the stack - a typical depth might be 5). A sensible minimum, however, might be

1. Current feedrate, and
2. Whether moves (and separately extrusion) are relative or absolute

M121: Pop

Recover the last state pushed onto the stack.

M122: Diagnose

Sending an M122 causes the RepRap to transmit diagnostic information, for eaxmple via a USB serial link.

M123: Tachometer value

Sending an M123 causes the RepRap to transmit filament tachometer values from all extruders.

M124: Immediate motor stop

Immediately stops all motors.

M126: Open Valve

Example: M126 P500

Open the extruder's valve (if it has one) and wait 500 milliseconds for it to do so.

M127: Close Valve

Example: M127 P400

Close the extruder's valve (if it has one) and wait 400 milliseconds for it to do so.

M128: Extruder Pressure PWM

Example: M128 S255

PWM value to control internal extruder pressure. S255 is full pressure.

M129: Extruder pressure off

Example: M129 P100

In addition to setting Extruder pressure to 0, you can turn the pressure off entirely. P400 will wait 100ms to do so.

M130: Set PID P value

Example: M130 P 0 S 8.0 # Sets heater 0 P factor to 8.0

Support	FiveD	Teacup	Sprinter	Marlin	Repetier
		yes		(See M13[01])	

Teacup can control multiple heaters with independent PID controls. For the default shown at https://github.com/Traumflug/Teacup_Firmware/blob/master/config.default.h, heater 0 is the extruder (P0), and heater 1 is the bed (P1).

Teacup's PID proportional units are in pwm/255 counts per quarter C, so to convert from counts/C, you would divide by 4. Conversely, to convert from count/qC to count/C, multiply by 4. In the above example, S=8 represents a Kp=8*4=32 counts/C.

M131: Set PID I value

Example: M131 P 1 S 0.5 # Sets heater 1 I factor to 0.5

Support	FiveD	Teacup	Sprinter	Marlin	Repetier
		yes		(See M30[01])	

Teacup's PID integral units are in pwm/255 counts per (quarter C*quarter second), so to convert from counts/qCqs, you would divide by 16. Conversely, to convert from count/qCqs to count/Cs, multiply by 16. In the above example, S=0.5 represents a Ki=0.5*16=8 counts/Cs.

M132: Set PID D value

Example: M132 P 0 S 24 # Sets heater 0 D factor to 24.0

Support	FiveD	Teacup	Sprinter	Marlin	Repetier
		yes		(See M30[01])	

Teacup's PID derivative units are in pwm/255 counts per (quarter degree per 2 seconds), so to convert from counts/C, you would divide by 4. Conversely, to convert from count/qC to count/C, multiply by 8. In the above example, S=24 represents a Kd=24*8=194 counts/(C/s).

M133: Set PID I limit value

Example: M133 P 1 S 264 # Sets heater 1 I limit value to 264

Support	FiveD	Teacup	Sprinter	Marlin	Repetier
		yes			

Teacup's PID integral limit units are in quarter-C*quarter-seconds, so to convert from C-s, you would multiply by 16. Conversely, to convert from qC*qs to C*s, divide by 16. In the above example, S=264 represents an integral limit of 16.5 C*s.

M134: Write PID values to EEPROM

Example: M134

Support	FiveD	Teacup	Sprinter	Marlin	Repetier
		yes			

M135: Set PID sample interval

Example: M135 S300

Set the PID to measure temperatures and calculate the power to send to the heaters every 300ms.

M136: Print PID settings to host

Example: M136 P1 # print heater 0 PID parameters to host

Support	FiveD	Teacup	Sprinter	Marlin	Repetier
		Debug			

M140: Bed Temperature (Fast)

Example: M140 S55

Set the temperature of the build bed to 55^{o}C and return control to the host immediately (*i.e.* before that temperature has been reached by the bed). There is an optional R field that sets the bed standby temperature: M140 S65 R40.

Support	FiveD	Teacup	Sprinter	Marlin	Repetier
		yes	yes	yes	yes

M141: Chamber Temperature (Fast)

Example: M141 S30

Set the temperature of the chamber to 30°C and return control to the host immediately (*i.e.* before that temperature has been reached by the chamber).

M142: Holding Pressure

Example: M142 S1

Set the holding pressure of the bed to 1 bar.

The holding pressure is in bar. For hardware which only has on/off holding, when the holding pressure is zero, turn off holding, when the holding pressure is greater than zero, turn on holding.

M143: Maximum hot-end temperature

Example: M143 S275

Set the maximum temperature of the hot-end to 275C

When temperature of the hot-end exceeds this value, take countermeasures, for instance an emergency stop. This is to prevent hot-end damage.

M144: Stand By Your Bed

Example: M144

Switch the bed to its standby temperature. M140 turns it back to its active temperature; no need for any arguments for that use of M140.

M160: Number of mixed materials

Example: M160 S4

This command has been superseded by the tool definition command M563 (see below).

Set the number of materials, N, that the current extruder can handle to the number specified. The default is 1.

When N >= 2, then the E field that controls extrusion requires N values separated by colons ":" after it like this:

```
M160 S4
G1 X90.6 Y13.8 E2.24:2.24:2.24:2.24:15.89
G1 X70.6 E0:0:0:42.4
G1 E42.4:0:0:0
```

The second line moves straight to the point (90.6, 13.8) extruding a total of 22.4mm of filament. The mix ratio for the move is 0.1:0.1:0.1:0.7.

The third line moves back 20mm in X extruding 42.4mm of filament.

The fourth line has no physical effect.

M190: Wait for bed temperature to reach target temp

Example: M190 S60

This will wait until the bed temperature reaches 60 degrees, printing out the temperature of the hot end and the bed every second.

Support	FiveD	Teacup	Sprinter	Marlin	Repetier
		obsolete, see M116	yes	yes	

M200 - Set filament diameter / Get Endstop Status

M200 Dm.mmm sets the filament diameter to m.mmm millimeters. It is used with 'volumetric calibration' and G-code generated for an ideal 1.128mm diameter filament, which has a volume of 1mm^3 per millimeter. The intention is to be able to generate filament-independent g-code. (See Triffid_Hunter's_Calibration_Guide#Optional:_Switch_to_volumetric_E_units and http://wooden-mendel.blogspot.com/2011/09/volumetric-stage-two.html for more information.)

Support	FiveD	Teacup	Sprinter	Marlin	Repetier
				yes	

M200 D0 or M200 D1.128 ; reset E multiplier to 1, since sqrt(1/pi)*2=1.128

See also Gcode#M119:_Get_Endstop_Status

Question: what does a firmware do with filament diameter? Has this an effect on how much an E command moves the extruder motor? --Traumflug 11:34, 14 October 2012 (UTC) Yes, Marlin uses this to set a 'volumetric_multiplier' by which the E-steps of a move are scaled in the planner. DaveX (talk) 16:44, 12 April 2014 (PDT)

M201 - Set max printing acceleration

Example: M201 X1000 Y1000 Z100 E2000

Sets the acceleration that axes can do in units/second^2 for print moves. For consistency with the rest of G Code movement this should be in units/(minute^2), but that gives really silly numbers and one can get lost in all the zeros. So for this we use seconds.

M202 - Set max travel acceleration

in units/s^2 for travel moves (M202 X1000 Y1000) Unused in Marlin!!

M203 - Set maximum feedrate

Example: M203 X6000 Y6000 Z300 E10000

Sets the maximum feedrates that your machine can do in mm/min.

M204 - Set default acceleration

S normal moves T filament only moves (M204 S3000 T7000) im mm/sec^2 also sets minimum segment time in ms (B20000) to prevent buffer underruns and M20 minimum feedrate

M205 - advanced settings

minimum travel speed S=while printing T=travel only, B=minimum segment time X= maximum xy jerk, Z=maximum Z jerk, E=maximum E jerk

M206: set home offset

Example: M206 X10.0 Y10.0 Z-0.4

The values specified are added to the endstop position when the axes are referenced. The same can be achieved with a G92 right after homing (G28, G161).

With Marlin firmware, this value can be saved to EEPROM using the M500 command.

A similar command is G10, aligning these two is subject to discussion.

M207: calibrate z axis by detecting z max length

Example: M207

After placing the tip of the nozzle in the position you expect to be considered Z=0, issue this command to calibrate the Z axis. It will perform a z axis homing routine and calculate the distance traveled in this process. The result is stored in EEPROM as z_max_length. For using this calibration method the machine must be using a Z MAX endstop.

This procedure is usually more reliable than mechanical adjustments of a Z MIN endstop.

NOTE: Marlin defines M207 as "set retract length S[positive mm] F[feedrate mm/min] Z[additional zlift/hop], stays in mm regardless of M200 setting"

M208: set axis max travel

Example: M208 X250 Y210 Z180

The values specified set the software limits for axis travel in the positive direction.

~~With Marlin firmware, this value can be saved to EEPROM using the M500 command.~~

With Duet-dc42 firmware, you can also use this command to specify software limits for axis travel in the negative direction, by adding parameter S1. The axis limits you set are also the positions assumed when an endstop is triggered.

NOTE: Marlin defines M208 as "set recover=unretract length S[positive mm surplus to the M207 S*] F[feedrate mm/sec]"

M209: enable automatic retract

Example: M209 S1

This boolean value S 1=true or 0=false enables automatic retract detect if the slicer did not support G10/11: every normal extrude-only move will be classified as retract depending on the direction.

M210: Set homing feedrates

Example: M210 X1000 Y1500

Set the feedrates used for homing to the values specified in mm per minute.

M220:set speed factor override percentage

Example: M220 S80

S<factor in percent>- set speed factor override percentage

M221: set extrude factor override percentage

Example: M221 S70

```
S<factor in percent>- set extrude factor override percentage
```

M226: Gcode Initiated Pause

Example: M226

Initiates a pause in the same way as if the pause button is pressed. That is, program execution is stopped and the printer waits for user interaction. This matches the behaviour of M1 in the NIST RS274NGC G-code standard (http://www.nist.gov/manuscript-publication-search.cfm?pub_id=823374) and M0 in Marlin firmware.

M227: Enable Automatic Reverse and Prime

Example: M227 P1600 S1600

P and S are steps.

"Reverse and Prime" means, the extruder filament is retracted some distance when not in use and pushed forward the same amount before going into use again. This shall help to prevent drooling of the extruder nozzle. Teacup firmware implements this with M101/M103.

M228: Disable Automatic Reverse and Prime

Example: M228

See also M227.

M229: Enable Automatic Reverse and Prime

Example: M229 P1.0 S1.0

P and S are extruder screw rotations. See also M227.

M230: Disable / Enable Wait for Temperature Change

Example: M230 S1

S1 Disable wait for temperature change S0 Enable wait for temperature change

M240: Start conveyor belt motor / Echo off

Example: M240

Support	FiveD	Teacup	Sprinter	Marlin	Repetier
		Debug: Echo off			

The conveyor belt allows to start mass production of a part with a reprap.

Echoing may be controlled in some firmwares with M111

M241: Stop conveyor belt motor / echo on

Example: M241

Support	FiveD	Teacup	Sprinter	Marlin	Repetier
		Debug: Echo on			

Echoing may be controlled in some firmwares with M111

M245: Start cooler

Example: M245

used to cool parts/heated-bed down after printing for easy remove of the parts after print

M246: Stop cooler

Example: M246

M280: Set servo position

M280 - set servo position absolute. P: servo index, S: angle or microseconds (Marlin)

M300: Play beep sound

Usage: M300 S<frequency Hz> P<duration ms>

Example: M300 S300 P1000

Play beep sound, use to notify important events like the end of printing. See working example on (http://www.3dprinting-r2c2.com/?q=content/seasons-greetings) R2C2 electronics.

M301: Set PID parameters - Hot End

```
Example: M301 P1 I2 D3 ; Marlin
Example: M301 P1 I2 D3 C5 H1 B20 W127 ; Duet
Example: M301 P1 I2 D3 T0.2 H1 B20 W127 ; Duet-dc42
```

Support	Duet	Teacup	Sprinter	Marlin	Repetier
	PIDCHW	(See M13[0-3])		PID	

Sets Proportional, Integral and Derivative values for hot end. The value C refers to an extrusion rate. The value T is the approximate additional PWM (on a scale of 0 to 255) needed to maintain each additional 1C temperature, used to preset the I-accumulator when switching from heater fully on/off to PID. Duet-dc42 firmware interprets a negative P term as indicating that bang-bag control should be used instead of PID (not recommended for the hot end, but OK for H0 which is the bed heater).

See also PID Tuning.

Alternate implementation

Example: M301 W125

See M130, M131, M132, M133 for Teacup's codes for setting the PID parameters.

M302: Allow cold extrudes

This tells the printer to allow movement of the extruder motor, when the hotend is not at printing temperature

Example: M302

M303: Run PID tuning

Hotend Usage: M303 S<temperature> C<cycles> Bed Usage: M303 E-1 C<cycles> S<temperature> Example: M303 C8 S175

Generate Proportional, Integral and Derivative values for the hotend or bed (E-1). Send the appropriate code and wait for the output to update the firmware.

	FiveD	Teacup	Sprinter	Marlin	Repetier
Support				PID	PID

M304: Set PID parameters - Bed

```
Example: M304 P1 I2 D3 ; set kP=3, kI=2, kD=3
Example: M301 P1 I2 D3 T0.7 H1 B20 W127 ; Duet-dc42 firmware
Example: M304          ; Report parameters
```

	FiveD	Teacup	Sprinter	Marlin	Repetier
Support				PID	

Sets Proportional, Integral and Derivative values for bed. Duet-dc42 firmware interprets a negative P term as indicating that bang-bag control should be used instead of PID.

See also PID Tuning.

M305: Set thermistor and ADC parameters

Sets the parameters for temperature measurement. Supported by Duet-dc42 firmware.

Example: M305 P1 T100000 R1000 B4200 H14 L-11

This tells the firmware that for heater 1 (P parameter: 0 = heated bed, 1 = first extruder) the thermistor 25C resistance (T parameter) is 100Kohms, the thermistor series resistance (R parameter) is 1Kohms, the thermistor beta (B parameter) is 4200, the ADC high end correction (H parameter) is 14 and the ADC low end correction (L parameter) is -11. All parameters other than P are optional. If only the P parameter is given, the existing values are displayed.

M400: Wait for current moves to finish

Finishes all current moves and and thus clears the buffer. That's identical to `G4 P0`.

Example: M400

M420: Set RGB Colors as PWM

Usage: M420 R<Red PWM (0-255)> E<Green PWM (0-255)> B<Blue PWM (0-255)>

Example: M420 R255 E255 B255

Set the color of your RGB LEDs that are connected to PWM-enabled pins. Note, the Green color is controlled by the E value instead of the G value due to the G code being a primary code that cannot be overridden.

M540: Set MAC address

Example: M540 P0xBE:0xEF:0xDE:0xAD:0xFE:0xED

Sets the MAC address (http://en.wikipedia.org/wiki/MAC_address) of the RepRap. This should be done before any other network commands. The MAC address is six one-byte hexadecimal numbers separated by colons. The 0x prefix is compulsory.

M550: Set Name

Example: M550 PGodzilla

Sets the name of the RepRap to (in this case) Godzilla. The name can be any string of printable characters except ';', which still means start comment.

M551: Set Password

Example: M551 Pmy-very-secret-word

On machines that need a password to activate them, set that password. The code 'P' is not part of the password. Note that as this is sent in clear it does not (nor is it intended to) offer a very high level of security. But on machines that are (say) on a network, it prevents idle messing about by the unauthorised. The password can contain any printable characters except ';', which still means start comment.

M552: Set IP address

Example: M552 P192.168.1.14

Sets the IP address of the RepRap machine to (in this case) 192.168.1.14. A restart may be required before the new IP address is used. If no P field is specified, this echoes the existing IP address configured.

M553: Set Netmask

Example: M553 P255.255.255.0

Sets the network mask of the RepRap machine to (in this case) 255.255.255.0. A restart may be required before the new network mask is used. If no P field is specified, this echoes the existing network mask configured.

M554: Set Gateway

Example: M554 P192.168.1.1

Sets the Gateway IP address of the RepRap machine to (in this case) 192.168.1.1. A restart may be required before the new gateway IP address is used. If no P field is specified, this echoes the existing Gateway IP address configured.

M555: Set compatibility

Example: M555 P1

For firmware that can do it, sets the firmware to a mode where its input and (especially) output behaves exactly like other established firmware. The value of the P argument is:

P value	Firmware
0	Native (i.e. whatever the firmware actually is)
1	RepRap_Firmware
2	Marlin
3	Teacup
4	Sprinter
5	Repetier

M556: Axis compensation

Example: M556 S100 X0.7 Y-0.2 Z0.6

Though with care and adjustment a RepRap can be set up with its axes at right-angles to each other within the accuracy of the machine, who wants to bother with care and adjustment when the problem can be solved by software? This tells software the tangents of the angles between the axes of the machine obtained by printing then measuring a test part. The S parameter (100 here) is the length of a triangle along each axis in mm. The X, Y and Z figures are the number of millimeters of the short side of the triangle that represents how out of true a pair of axes is. The X figure is the error between X and Y, the Y figure is the error between Y and Z, and the Z figure is the error between X and Z. Positive values indicate that the angle between the axis pair is obtuse, negative acute.

M557: Set Z probe point

Example: M557 P1 X30 Y40.5

Set the points at which the bed will be probed to compensate for its plane being slightly out of horizontal. The P value is the index of the point (indices start at 0) and the X and Y values are the position to move extruder 0 to to probe the bed. An implementation should allow a minimum of three points (P0, P1 and P2). This just records the point coordinates; it does not actually do the probing. See G32.

M558: Set Z probe type

Example: M558 P0

A Z probe may be a switch (the default) an IR proximity sensor, or some other device. This selects which to use. P0 gives a switch. P1 gives an unmodulated IR probe, or any other probe type that emulates an unmodulated IR probe (probe output is an analog signal that rises with decreasing nozzle height above the bed). If there is a control signal to the probe, it is driven high when the probe type is P1. P2 specifies a modulated IR probe, where the modulation is commanded directly by the main board firmware using the control signal to the probe. P3 selects an alternative Z probe by driving the control signal to the probe low. See also G31 and G32.

M559: Upload configuration file

Example: M559

If the RepRap supports it, this uploads a file that is run on re-boot to configure the machine. This file usually is a special G Code file. After sending M559, the file should be sent, ending with an M29 (q.v.).

M560: Upload web page file

Example: M560

For RepRaps that have web support and that can be driven by a web browser, this uploads the file that is the control page for the RepRap. After sending M560 the file (usually an HTML file) should be sent, terminated by the string

```
<!-- **EoF** -->
```

. Clearly that string cannot exist in the body of the file, but can be put on the end to facilitate this process. This should not be too serious a restriction...

M561: Set Identity Transform

Example: M561

This cancels any bed-plane fitting as the result of probing (or anything else) and returns the machine to moving in the user's coordinate system.

M562: Reset temperature fault

Example: M562 P2

Reset a temperature fault on heater/sensor 2. If the RepRap has switched off and locked a heater because it has detected a fault, this will reset the fault condition and allow you to use the heater again. Obviously to be used with caution. If the fault persists it will lock out again after you have issued this command. P0 is the bed; P1 the first extruder, and so on.

M563: Define a tool

Example: M563 P3 D0:5:6 H1:3

Tools are usually (though not necessarily) extruders. The P field specifies the tool number. The D field specifies the drive(s) used by the tool - in this case drives 0, 5 and 6. Drive 0 is the first drive in the machine after the movement drives (usually X, Y and Z). If there is no D field the tool has no drives. The H field specifies the tool's heaters - in this case heaters 1 and 3. Heater 0 is usually the hot bed (if any) so the first extruder heater is usually 1. If there is no H field the tool has no heaters.

Tools are driven using multiple values in the E field of G1 commands, each controlling the corresponding drive in the D field above, as follows:

```
G1 X90.6 Y13.8 E2.24:2.24:15.89
G1 X70.6 E0:0:42.4
```

The first line moves straight to the point (90.6, 13.8) extruding a total of 2.24mm of filament from both drives 0 and 5 and 15.98mm of filament from drive 6.

The second line moves back 20mm in X extruding 42.4mm of filament from drive 6.

Normally an M563 command is immediately followed by a G10 command to set the tool's offsets and temperatures.

It is permissible for different tools to share some (or all) of their drives and heaters. So, for example, you can define two tools with identical hardware, but that just operate at different temperatures.

M564: Limit axes

Example: M564 S0

Allow moves outside the print volume, or not. If the S parameter is 0, then you can send G codes to drive the RepRap outside its normal working volume, and it will attempt to do so. User beware... If you set the S parameter to 1 then the RepRap will not think outside the box. The default behaviour is S = 1.

M565: Set Z probe offset

Example: M565 X3 Y4.5 Z-2.37

Set the offset from the extruder tip to the probe position. The X, Y and Z values are the delta between the extruder and the actual trigger position of the probe. If the probe trigger point is below the extruder (typical) the Z offset will be negative. This just records the point offset; it does not actually do the probing. See G32.

M566: Set minimum speeds

Example: M566 X20 Y20 Z2 E10

Sets the speeds in mm/minute that axes can do from a standing start. If an accelerating algorithm starts a move with a zero velocity then accelerates from that, it can give problems when the zero initial velocity is used to calculate a timestep between stepper pulses at the begining: the timestep ends up being infinite... So most systems have initial small velocities to start at. This sets them.

M567: Set tool mix ratios

Example: M567 P2 E0.1:0.2:0.1:0.6

The example sets the mix ratio for tool 2 (the P value). When mixing is then turned on (see M568), only single E values need to be sent on a G1 command (any extra E values will be ignored, but are not illegal):

G1 X20 E1.3

This will move to X=20 extruding a total length of filament of 1.3mm. The first drive of tool 2 will extrude 0.1*1.3mm, the second 0.2*1.3mm and so on. The ratios don't have to add up to 1.0 - the calculation done is as just described. But it is best if they do.

See also M568.

M568: Turn off/on tool mix ratios

Example: M568 P2 S0

Turn on/off automatic mix ratios for tool 2. If the S parameter is 0 mixing is turned off; if it is non-zero it is turned on.

After turning off command G1 instructions must send as many E values as the tool has drives:

G1 X20 E0.2:0.4:0.166:0.3

The off state is the default.

M569: Set axis direction values

Example: M569 P0 S1

Set the control value for the drive specified by P that sends it forwards to the given value in the S field. After sending the example, sending a 1 to X (drive 0) will make it go forwards, sending a 0 will make it go backwards. Obviously to be used with extreme caution...

M570: Set heater timeout

Example: M570 S120

After a heater has been switched on, wait 120 seconds for it to get close to the set temperature. If it takes longer than this, flag a heater fault.

M665: Set delta configuration (Marlin)

Example: M660 L250 R160 S200

Set the delta calibration variables. L = diagonal rod length, R = delta radius, S = segments per second.

M906: Set motor currents

Example: M906 X300 Y500 Z200 E350

Sets the currents to send to the stepper motors for each axis. The values are in milliamps.

M998: Request resend of line

Example: M998 P34

Request a resend of line 34. In some implementations the input-handling code overwrites the incomming G Code with this when it detects, for example, a checksum error. Then it leaves it up to the GCode interpreter actually to request the resend.

M999: Restart after being stopped by error

Example: M999

T: Select Tool

Example: T1

Select tool (or in older implementations extruder) number 1 to build with.

The sequence followed is:

1. Set the current tool to its standby temperatures specified by G10 (see above),
2. Set the new tool to its operating temperatures specified by G10 and wait for **all** temperatures to stabilise,
3. Apply any X, Y, Z offset for the new tool specified by G10,
4. Use the new tool.

Selecting a non-existent tool (100, say) just does Step 1. above. That is to say it leaves all tools in their standby state. You can, of course, use the G10 command beforehand to set that standby temperature to anything you like.

Note that you may wish to move to a parking position *before* executing a T command in order to allow the new extruder to reach temperature while not in contact with the print. It is acceptable for the firmware to apply a small offset [by convention (-1mm x tool-number) in Y] to the current position when the above sequence is entered to allow temperature changes to take effect just away from the parking position. Any such offset must, of course, be undone when the procedure finishes.

If the Z value changes in the offsets and the tool moves up, then the Z move is made before the X and Y moves. If Z moves down, X and Y are done first.

Some implementations (e.g. RepRapFirmware) allow you to specify tool-change G Code macros. There are normally three specified (any of which can contain no commands if desired) that execute in this order:

1. Actions to do with the old tool before it is released - macro name: **tfreeN.g** where N is the tool number;
2. (Old tool is released);
3. Actions to do with the new tool before it is selected - macro name: **tpreN.g** where N is the tool number;
4. (New tool is selected); and
5. Actions to do with the new tool after it is selected - macro name: **tpostN.g** where N is the tool number.

With such implementations there is no wait for temperature stabilisation. That can be achieved by an M116 in any of the macros, of course.

After a reset tools will not start heating until they are selected. You can either put them all at their standby temperature by selecting them in turn, or leave them off so they only come on if/when you first use them. The M0, M1 and M112 commands turn them all off. You can, of course, turn them all off with the M1 command, then turn some back on again. Don't forget also to turn on the heated bed (if any) if you use that trick.

Tool numbering may start at 0 or 1, depending on the implementation. Some implementations (those that use the M563 command to define tools) allow the user to specify tool numbers, so with them you can have tools 17, 99 and 203 if you want. Negative numbers are not allowed.

Proposed SCARA calibration codes (Morgan)

In order to ease calibration of Reprap Morgan, the following M-codes are used to set the machine up Implemented in qharley/Marlin armlevel branch.

M360 : Move to Theta 0 degree position

The arms move into a position where the Theta steering arm is parallel to the top platform edge. The user then calibrates the position by moving the arms with the jog buttons in software like pronterface until it is perfectly parallel. Using M114 will then display the calibration offset that can then be programmed into the unit using M206 (Home offset) X represents Theta.

M361 : Move to Theta 90 degree position

Theta move to 90 degrees with platform edge. User calibrates by using jog arms to place exactly 90 degrees. Steps per degree can then be read out by using M114, and programmed using M92. X represents Theta. Program Y (Psi) to the same value initially. Remember to repeat M360 after adjusting steps per degree.

M362 : Move to Psi 0 degree position

Arms move to Psi 0 degree. Check only after other Theta calibrations

M363 : Move to Psi 90 degree position

Arms move to Psi 90 degree. Check only after other Theta calibrations

M364 : Move to Psi + Theta 90 degree position

Move arms to form a 90 degree angle between the inner and outer Psi arms. Calibrate by moving until angle is exactly 90 degree. Read out with M114, and calibrate value into Home offset M206. Psi is represented by Y.

M365 : SCARA scaling factor

Adjust X Y and Z scaling by entering the factor. 100% scaling (default) is represented by 1

M370 : Morgan manual bed level - clear map

Clear the map and prepare for calibration Usage:

```
M370
M370 X<divisions> Y<divisions>
```

Without parameters is defaults to X5 Y5 (25 calibration points) When specifying parameters, uneven numbers are recommended.

M371 : Move to next calibration position

Move to the next position for calibration. User moves the bed towards the hotend until it just touches

M372 : Record calibration value, and move to next position

The position of the bed is recorded and the machine moves to the next position. Repeat until all positions programmed

M373 : End bed level calibration mode

M375 : Display matrix

Display the bed level calibration matrix

Store the calibration to EEPROM using M500

Proposed EEPROM configuration codes

BRIEFLY: each RepRap has a number of physical parameters that should be persistent, but easily configurable, such as extrusion steps/mm, various max values, etc. Those parameters are currently hardcoded in the firmware, so that a user has to modify, recompile and re-flash the firmware for any adjustments. These configs can be stored in MCU's EEPROM and modified via some M-codes. Please see the detailed proposal at M-codes for EEPROM config. (*This is proposed by --AlexRa on 11-March-2011. There is currently no working implementation of the proposed commands*).

Marlin uses these codes to manipulate EEPROM values.

Sprinter has implemented the following commands to manipulate EEPROM Commit message (https://github.com/kliment/Sprinter/commit/4b1b0f1d96d2be2ed3941095f40a5c2d2bbb943d) .

Teacup uses codes M130-M136 to set, read, and save some parameters.

M500: stores paramters in EEPROM

M501: reads parameters from EEPROM

If you need to 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.

M503: Print settings

Replies from the RepRap machine to the host computer

All communication is in printable ASCII characters. Messages sent back to the host computer are terminated by a newline and look like this:

xx [line number to resend] [T:93.2 B:22.9] [C: X:9.2 Y:125.4 Z:3.7 E:1902.5] [Some debugging or other information may be here]

xx can be one of:

ok

rs

!!

ok means that no error has been detected.

rs means resend, and is followed by the line number to resend.

!! means that a hardware fault has been detected. The RepRap machine will shut down immediately after it has sent this message.

The **T:** and **B:** values are the temperature of the currently-selected extruder and the bed respectively, and are only sent in response to M105. If such temperatures don't exist (for example for an extruder that works at room temperature and doesn't have a sensor) then a value below absolute zero (-273^{o}C) is returned.

C: means that coordinates follow. Those are the **X: Y:** etc values. These are only sent in response to M114 and M117.

The RepRap machine may also send lines that look like this:

// This is some debugging or other information on a line on its own. It may be sent at any time.

Such lines will always be preceded by //.

The most common response is simply:

ok

When the machine boots up it sends the string

start

once to the host before sending anything else. This should not be replaced or augmented by version numbers and the like. M115 (see above) requests those.

All this means that every line sent by RepRap to the host computer except the start line has a two-character prefix (one of **ok**, **rs**, **!!** or //). The machine should never send a line without such a prefix.

Exceptions: Marlin 1.0.0 Gen6 Firmware does not follow the two character rule. 'rs' is actually 'Resend' and '!!' is 'Error'. Example Lines:

- Error: Line Number is not current line + 1. Last Line: 7
- Resend: 8
- Writing to File: print.gco
- Done saving file.
- File opened:print.gco Size:22992
- File selected

When in the code base did this change take place and what other firmwares are affected?

Proposal for sending multiple lines of G-code

So far, this is a proposal, open for discussion.

Problem to solve

Each line of G-code sent from the host to the controller is answered with an **ok** before the next line can be sent without locking communcations up. This makes operations very slow, as the usual USB-TTL converters and probably also the host's operating system drivers come with substantial latency, often 10 milliseconds.

For more details on this proposal, and some suggested solutions, and comments please see GCODE_buffer_multiline_proposal

Alternatives to G-code

Main article: Firmware/Alternative#alternatives to G-code

Several people have suggested using STEP-NC or some other control language; or perhaps designing a completely new control language.