Gcode Marlin FR

G-code

Cette page tente de décrire le 'G-code' que les firmwares Reprap utilisent et comment ils fonctionnent. La cible principale est la fabrication additive en utilisant le processus FFF. Codes pour les mouvements de la tête d'impression suivent le [http://www.nist.gov/manuscript-publication-search.cfm?pub_id=823374 NIST RS274NGC norme G-code], de sorte que les firmwares Reprap sont tout à fait utilisables pour le fraisage CNC et autres applications similaires. Voir aussi sur [https://en.wikipedia.org/wiki/G-code G-Code article de Wikipedia].

Il ya plusieurs façons de préparer un Gcode pour une imprimante. La première consiste à utiliser un trancheur comme Slic3r, Skeinforge ou Cura. Ces programmes prennent un modèle CAO et le découpent en couches. Les trancheurs sont la meilleure façon de passer d'un modèle 3D en fichier gcode imprimable, mais l'utilisateur sacrifie une certaine flexibilité lors de leur utilisation. Une autre option pour la génération gcode est d'utiliser une bibliothèque de niveau inférieur comme mecode. Les bibliothèques comme mecode vous donnent un contrôle précis sur la trajectoire de l'outil, et ainsi sont utiles si vous avez une impression complexe qui ne convient pas pour le tranchage natïf. La dernière option est de simplement écrire le Gcode vous-même. Cela peut être le meilleur choix si vous avez juste besoin de courir quelques lignes de test lors de l'étalonnage de votre imprimante.

Introduction

Un morceau de fichier Gcode d'une imprimante 3D Reprap peut ressembler à ça:

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

Le Gcode peut être stocké dans un fichier sur une carte SD, votre disque dur ou tout autre support. L'extension du fichier Gcode peut être '.g', '.gco' or '.gcode'. Pour les BFB/RapMan l'extension est '.bfb'. Le fichier Gcode stocké et/ou créé par un trancheur peut également rassembler à ça:

G92 E0 G28 G1 F1500 G1 X2.0 Y2.0 F3000 G1 X3.0 Y3.0

La signification de tous ces symboles et chiffres (et plus) est expliquée ci-dessous.

Les trancheurs peuvent éventuellement ajouter des scripts de gcode au début et à la fin de leur fichier afin d'effectuer des actions spécifiques avant et / ou après l'impression tels que : l'autoleveling, le chauffage / refroidissement du lit et hotend, effectuer un mouvement pour essuyer la buse grâce à un servomoteur, mise en route de l'alimentation Ps_on (ATX). Plus d'infos sur les pages Start GCode routines et End GCode routines.

Pour savoir quels GCode spécifique(s) sont mis en œuvre dans cette page, il y a de petites tables attachées à la description des commandes, comme celle-ci:

fived={{yes | teacup=automatic | sprinter=no | marlin=partial | repetier=experimental |

G0 & G1: Move

smoothie=deprecated }}

fived={{yes | teacup=yes | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=yes | bfb=yes | machinekit=yes | makerbot=yes | grbl=yes | redeem=yes }}

Legende: ; yes : Le Gcode est complètement supporté par le firmware. ; partial or experimental : Le Gcode peut fonctionner après manipulation. Souvent, il est nécessaire de consulter la branche de code source pour le firmware (généralement stocké dans une branche différente) ou pour inverser les commutateurs de configuration sur la carte mère. ; automatic : Le firmware gère cet GCode automatiquement, donc il n'y a pas besoin d'envoyer la commande. Un exemple est alimentation on / off (M80 / M81) dans le firmware Teacup. ; ??? : On ne sait pas si le firmware prend en charge ce GCode. Vous devez le tester vous même. ; no : Le firmware ne support pas le Gcode ; deprecated : Le firmware a retiré le support de ce Gcode. L'auteur du firmware a sans doute retiré ce Gcode en le remplaçant par un autre.

* G0 : Déplacement rapide * G1 : Déplacement Linéaire

; Usage : G0 Xnnn Ynnn Znnn Ennn Fnnn Snnn : G1 Xnnn Ynnn Znnn Ennn Fnnn Snnn ; Paramètre : Tous les paramètres ne nécessitent pas d'être utilisés, mais au moins un doit être utilisé : 'Xnnn' L'axe X bouge sur sa position : 'Ynnn' L'axe Y bouge sur sa position : 'Znnn' L'axe Z bouge sur sa position : 'Ennn' Le Moteur de l'extrudeur fait avancer le filament de nnn millimètres : 'Fnnn' La vitesse en millimètres par minute d'un ou des axes : 'Snnn' Test pour vérifier si un endstop a été déclenché ('S1' pour vérifier, 'S0' pour ignorer, 'S2' voir la note , par défaut si 'S0')[1] ; Exemples : G0 X12 (Déplacer l'axe X de 12mm) : G0 F1500 (Régler la vitesse à 1500mm/minute) : G1 X90.6 Y13.8 E22.4 (Déplacer l'axes X de 90.6mm l'axeY de 13.8mm en extrudant 22.4mm de matière)

Fields

Un GCode RepRap est une liste de champs qui sont séparés par des espaces blancs ou des sauts de ligne. Un champ peut être interprété comme une commande, paramètre, ou pour tout autre but spécial. Il se compose d'une lettre directement suivi d'un nombre, ou peut être seulement une lettre autonome (Flag). La lettre donne des informations sur le sens du champ (voir la liste ci-dessous dans cette section). Les nombres peuvent être `entiers (128)` ou `nombres fractionnaires (12.42)`, en fonction du contexte. Par exemple, une coordonnée X peut prendre entiers (`'x175'`) ou fractionels (`'X17.62'`), mais sélectionner un extrudeur par un chiffre tel que 3.14 aurait aucun sens. Dans cette description, les numéros dans les domaines sont représentés par `'nnn'` comme un espace réservé.

Cependant, dans l'exemple ci-dessus, nous avons mis une avance de 1500 mm / minute sur la ligne 1, puis déplacé les axes comme demandé mais à 3000 mm / minute. L'extrusion va accélérer avec le X et le mouvement Y, donc tout reste synchronisé.

La spécification RepRap traite l'avance comme simplement une autre variable (comme X, Y, Z et E) pour être interpolées linéairement. Cela donne un contrôle complet sur l'accélération et la décélération de la tête d'impression de manière à ce que le firmware assure que tout se déplace ensemble.[3]

Pour inverser l'extrudeuse par une quantité donnée (par exemple pour réduire sa pression interne pendant qu'il fait un mouvement (en X,Y ou Z) afin qu'il ne dribble pas) il suffit d'utiliser G0 ou G1 d'envoyer une valeur de E qui est inférieure à la longueur actuellement extrudé .

Les Firmware RepRap ont la même spécificité pour les commandes G0 and G1 .[2]

<pre> 1. G1 F1500 2. G1 X50 Y25.3 E22.4 </pre>

Dans l'exemple ci-dessus, nous avons mis l'avance à 1500mm / minute sur la ligne 1, puis déplacé l'axe X de 50 mm, l'axe Y de 25.3mm et extrudé 22.4mm de filament.

<pre> 1. G1 F1500 2. G1 X50 Y25.3 E22.4 F3000 </pre>

{| class="wikitable"

G2 & G3: Controle des mouvements des Cercles

fived={{no | teacup=no | sprinter=yes | marlin=yes[1] | repetier=yes | smoothie=yes | reprapfirmware=no | bfb=no | machinekit=yes | grbl=yes }}

; Usage : G2 Xnnn Ynnn Innn Jnnn Ennn Fnnn (Cercle dans le sens des aiguilles d'une montre) : G3 Xnnn Ynnn Innn Jnnn Ennn Fnnn (Cercle dans le sens inverse des aiguilles d'une montre) ; Parameters : 'Xnnn' L'axe X bouge sur sa position : 'Ynnn' L'axe Y bouge sur sa position : 'Innn' Le point X part de sa position actuelle X afin de maintenir une distance constante à partir de : 'Jnnn' Le point Y part de sa position actuelle X afin de maintenir une distance constante à partir de : 'Ennn' Le Moteur de l'extrudeur fait avancer le filament de nnn millimètres :

'Fnnn' La vitesse en mm/minute d'un ou des axes ; Examples : G2 X90.6 Y13.8 I5 J10 E22.4 (Déplacement dans le sens des aiguilles d'une montre de ce point à ce point(X=90.6,Y=13.8), avec un point central à (X=current_X+5, Y=current_Y+10), en extrudant 22.4mm de matière entre le début et la fin du mouvement des axes) : G3 X90.6 Y13.8 I5 J10 E22.4 (Déplacement dans le sens inverse des aiguilles d'une montre de ce point à ce point(X=90.6,Y=13.8), avec un point central à (X=current_X+5, Y=current_Y+10), en extrudent 22.4mm de matière entre le début et la fin du mouvement des axes)

Notes

Dans le Firmware Marlin cette commande n'est pas implanté pour les imprimantes 'DELTA' et 'SCARA' .

G4: Attente

fived={{yes | teacup=yes | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=yes | bfb=yes | machinekit=yes | makerbot=yes | grbl=yes | redeem=yes }}

; Paramètres : 'Pnnn' Temps à attendre, en millisecondes : 'Snnn' Temps à attendre, en secondes (Seulement pour Marlin et Smoothieware) ; Exemple : G4 P200

Dans ce cas, rien ne se passe pendant 200 millisecondes. Pendant les pauses, l'état de la machine (par exemple, les températures de ses hotend) sera toujours préservé et contrôlé.

Pour Marlin et Smoothie, le paramètre d'attente “S” est en secondes, et le paramètre “P” est en millisecondes.

G20: Définir les unités en Pouces

fived={{yes | teacup=yes | sprinter=yes | marlin=no | repetier=yes | smoothie=yes | reprapfirmware=yes | bfb=no | machinekit=yes | grbl=yes }}

Example: G20

À partir de cette commande, les unités sont en pouces.

G21: Définir les unités en Millimètres

fived={{yes | teacup=yes | sprinter=yes | marlin=no | repetier=yes | smoothie=yes | reprapfirmware=yes | bfb=yes | machinekit=yes | grbl=yes | redeem=yes }}

Example: G21

À partir de cette commande, les unités sont en millimètre. (Le millimètre est l'unité par défaut sur les RepRap.)

G28: Déplacement aux origines (Homing)

fived={{yes | teacup=yes | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=yes | bfb=yes | machinekit=yes | makerbot=yes | grbl=yes | redeem=yes }}

; Paramètres : `Ces Commandes peuvent être utilisées sans paramètres additionnels.` : 'X' Déplacement de l'axe X vers son origine : 'Y' Déplacement de l'axe Y vers son origine : 'Z' Déplacement de l'axe Z vers son origine ; Exemples : G28 `(Déplacement de tous les axes vers leurs origines)` : G28 X Z `(Déplacement seulement des axes X et Y vers leurs origines)`

Lorsque le firmware RepRap reçoit cette commande, il déplace le ou les axes vers leur(s) butées aussi rapidement que possible, puis les reculent d'un millimètre et les déplacent lentement vers les points d'activation de butée de position zéro pour augmenter la précision de position. Ce processus est également connu sous le nom de “`Homing`”.

Si vous ajoutez une coordonnée, elle sera ignorée. Par exemple,

```
G28&amp;nbsp;Z0.00
```

le résultat sera le même que

```
G28&amp;nbsp;Z
```

.

G29: Autolevel

fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=yes | smoothie=no | reprapfirmware=no, see G32 | bfb=no | machinekit=yes | redeem=yes }}

Example: G29

Déplace l'axe Z sur 3 points ou plus pour calculer la planéité du plateau. La commande G28 est à exécuter avant le G29.

G90: Position Absolue

fived={{yes | teacup=yes | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=yes | bfb=yes | machinekit=yes | grbl=yes | redeem=yes }}

Example: G90

Toutes les coordonnées éxécutées à partir de maintenant sont en rapport à l'origine de la machine. (la position absolue est l'unité par défaut sur les RepRap.)

G91: Position Relative

fived={{yes | teacup=yes | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=yes | bfb=no | machinekit=yes | grbl=yes | redeem=yes }}

Example: G91

Toutes les coordonnées à partir de maintenant sont liés à la dernière position.

G92: Définir la Position

fived={{yes | teacup=yes | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=yes | bfb=yes | machinekit=yes | makerbot=yes | grbl=yes | redeem=yes }}

; Parameters : `Ces Commandes peuvent être utilisées sans paramètres additionnels.` : 'Xnnn' Définit la nouvelle position de l'axe X : 'Ynnn' Définit la nouvelle position de l'axe Y : 'Znnn' Définit la nouvelle position de l'axe Z : 'Ennn' Définit la nouvelle position de l'extrudeur ; Exemple : G92 X10 E90

Permet la programmation du point zéro absolu, en réinitialisant la position actuelle aux valeurs spécifiées. Ici le X de la machine passe en coordonnée 10, et l'extrudeur à 90. Aucun mouvement physique se produira.

Un G92 sans coordonnées définira tous les axes à 0.

M-commandes

M0: Arret optionnel

fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=no | smoothie=no | reprapfirmware=no | bfb=no | machinekit=no | makerbot=no }}

Permet de mettre en pause l'impression à un moment pré-défini.

Exemple de code pour insérer une pièces métallique dans une impression, pour un décapsuleur par exemple. Et d'ensuite d'aller à Z10 et à Y200. Code à placer avant que l'imprimante se déplace à la prochaine couche dans le gcode.

- G0 Z10

- G0 Y200

- M0

Il suffit de relancer l'impression ensuite normalement.

M18: Arret de tous les moteurs

fived={{no | teacup=use M2 | sprinter=no | marlin=call M84 | repetier=no | smoothie=yes | reprapfirmware=yes | bfb=no | machinekit=no | makerbot=yes }}

Exemple: M18

Arrête tous les moteurs et permet de les tourner librement à la main'

le firmware Marlin permet de désactiver un des moteurs pas à pas. Par exemple, M18 X permet de désactiver les moteurs X.

M80: ATX Power On

fived={{no | teacup=automatic | sprinter=yes | marlin=yes | repetier=yes | smoothie=no | reprapfirmware=dc42 | bfb=no | machinekit=no }}

Example: M80

Active l'alimentation de l'ATX partir du mode veille au mode opérationnel.

M81: ATX Power Off

fived={{no | teacup=automatic | sprinter=yes | marlin=yes | repetier=yes | smoothie=no | reprapfirmware=dc42 | bfb=no | machinekit=no }}

Example: M81

Éteint l'alimentation ATX.

M82: Définir le mode Absolu pour l'extrudeur

fived={{no | teacup=yes | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=yes | bfb=no | machinekit=no }}

Exemple: M82

Définit la position de l'extrudeur comme position absolue

Ce mode est par défaut pour repetier.

M83: Définir le mode relatif pour l'extrudeur

fived={{no | teacup=yes | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=yes | bfb=no | machinekit=no }}

Example: M83

Défini la position de l'extrudeur comme position relatif du dernier point.

M92: Définir les steps/mm

fived={{no | teacup=no | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=yes | bfb=no | machinekit=no }}

Exemple (Sprinter et Marlin): M92 X<newsteps>

Permet la programmation des pas par unité (généralement mm) de l'axe (ici X). Le firmware se remet par défaut lors de la réinitialisation, à moins de sauvegarder les paramètres dans l'EEPROM (M500 à Marlin) ou dans le fichier de configuration (config.g dans RepRapFirmware). Cette commande est très utile pour l'étalonnage.

M104: Définir la température de l'extudeur

fived={{yes | teacup=yes | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=yes | bfb=yes | machinekit=yes | makerbot=yes | redeem=yes }}

; Paramètres : ' Snnn' Température cible ; Exemple : M104 S190

Règle la température de l'extrudeuse actuelle à 190°C et le maintien à cette température.

M105: Retourner la température de l'extrudeur et du plateau chauffant

fived={{yes | teacup=yes | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=yes | bfb=no | machinekit=no | redeem=yes }}

Example: M105

Renvoie la température actuelle de l'extrudeur et du plateau chauffant en degrés Celsius. La température est retournée sur l'ordinateur connecté. Par exemple, le retour à l'ordinateur connecté ressemble à ça : ok T:201 B:117

Extension/généralisation du M105 to be considered using S1 parameter as noted in Pronterface I/O Monitor

Dans Repetier, vous pouvez ajouter X0 pour récupérer la valeur brute si nécessaire : <pre> M105 X0 ==⇒ 11:05:48.910 : T:23.61 /0 @:0 T0:23.61 /0 @0:0 RAW0:3922 T1:23.89 /0 @1:0 RAW1:3920 </pre>

Extension Duet-dc42

Le firmware Duet-dc42 retourne une réponse au format JSON si le paramètre S2 ou S3 est ajouté dans la commande. C'est utilisé par le panneau de contrôle tactile. De plus, il est possible d'ajouter le

paramètre Rnn, où nn est le numéro de séquence de la dernière réponse reçue par le client.

La réponse contient un objet JSON, sans aucun objet ou tableau imbriqué, suivi par un caractère de saut de ligne. C'est similaire à l'objet retourné à l'interface web via les requêtes de statut, mais quelques champs sont omis. Voici un exemple de réponse lorsque le paramètre S2 est fourni:

{“status”:“I”,“heaters”:[25.0,29.0,28.3],“active”:[-273.1,0.0,0.0],“standby”:[-273.1,0.0,0.0],“hstat”:[0,2,1],“pos”:[-11.00,0.00,0.00],“extr”:[0.0,0.0],“sfactor”:100.00,

```
"efactor":[100.00,100.00],"tool":1,"probe":"535","fanRPM":0,"homed":[0,0,0],
"fraction_printed":0.572}
```

La signification de ces champs est la suivante:

status: I=inactif, P=impression en cours depuis la carte SD, S=stoppé(i.e. nécessite un reset), C=exécution du fichier de configuration, A=en pause, D=attente de mise en pause, R=reprise, B=occupé (exécution de macro en cours) heaters: températures actuelles des systèmes de chauffe, numérotés comme sur la machine(typiquement, le heater 0 est le plateau chauffant) active: températures actives des systèmes de chauffe standby: températures de standby des systèmes de chauffe hstat: état des systèmes de chauffe, 0=off, 1=standby, 2=actif, 3=erreur pos: positions X, Y et Z de l'extrudeur extr: positions des extrudeurs (axe E) sfactor: facteur de vitesse actuel (voir commande M220) efactor: facteur d'extrusion actuel (voir commande M221) tool: numéro de l'outil actuel. Typiquement, zéro signifie `aucun outil sélectionné` probe: valeur de la sonde Z fanRPM: vitesse du ventilateur de refroidissement en RPM homed: état de mise à zéro des axes X, Y et Z (ou des piliers sur une delta). 0=axe non mis à zéro, donc sa position n'est pas fiable; 1=axe mis à zéro, sont sa position est fiable. fraction_printed: la fraction du fichier en train d'être imprimée, qui a été lue et au moins traitée en partie. message: le message qui doit être affiché à l'écran (présent seulement d'il y a un message à afficher) timesLeft: un tableau des temps d'impression restant estimés (en secondes), calculés selon différentes méthodes. Ils sont actuellement basés sur la quantité du fichier lue, la longueur de filament consommé, et ne nombre de couches déjà imprimées. Champ seulement présent si on imprime depuis une carte SD. seq: numéro de séquence de la dernière réponse G-Code ou message d'erreur. Seulement présent si le paramètre R est fourni, et si le numéro de séquence est plus grand. resp: la dernière réponse G-Code ou message d'erreur. Seulement présent si le paramètre R est fourni, et si le numéro de séquence est plus grand.

Quand le paramètre S3 est utilisé, la réponse contient ces champs, ainsi que quelques champs additionnels qui ne changent généralement pas, et donc n'ont pas besoin d'être traités aussi souvent. Ces champs additionnels comprennent:

myName: le nom de l'imprimante geometry: “cartesian”, “delta”, “corexy, “corexz” etc.

La réponse devrait fournir les champs dans cet ordre. D'autres implémentations peuvent omettre certains champs, ou en ajouter d'autres.

M106: Allumer les ventilateurs

fived={{yes | teacup=yes | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=yes | bfb=yes | machinekit=yes }}

; Paramètres : 'Snnn' Vitesse ; Exemple : M106 S127

Allume les ventilateurs à la moitié de leur vitesse maximale.

Le paramètre obligatoire 'S' indique la valeur PWM (0-255). M106 S0 Eteint les ventilateurs. Dans certaines implémentations le PWM est spécifié par une fraction réelle : M106 S0.7

M106 dans le firmware Duet

Duet-dc42 firmware support un paramètre optionnel I. Si ce paramètre est présent et plus grand que 0, la sortie du ventilateur de refroidissement est inversé. Cela rend la sortie du ventilateur de refroidissement adapté pour alimenter l'entrée d'un ventilateur PWM à 4 fils via une diode. Si le paramètre est présent et à 0 ou négatif, la sortie n'est pas inversée. Si le paramètre n'est pas présent, l'état d'inversion/non-inversion est inchangé. La valeur par défaut au démarrage est pas inversée.

Si le paramètre 'R' est indiqué lors de l'utilisation Duet - zpl firmware (0,96 g +), la dernière valeur connue de ventilation sera réglée. Si le paramètre 'S' est indiqué avec 'R', le firmware n'effacera pas la dernière valeur connue de ventilation. Cela peut être utile pour les fichiers de macro de changement d'outil.

M107: Eteindre les ventilateurs

fived={{yes | teacup=no | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=yes | bfb=yes | machinekit=yes }}

Déprécié dans le Teacup firmware. Utilisez M106 S0 plutôt.

M108: Définir la vitesse d'extrusion

fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=no | reprapfirmware=no | bfb=yes | machinekit=no }}

Défini la vitesse du moteur d'extrusion (Déprécié dans le firmware FiveD, voir M113)

M109: Définir la température de l'extrudeur et attendre

fived={{yes | teacup=not needed | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=yes | bfb=no | machinekit=yes | makerbot=yes | redeem=yes }}

; Paramètres : 'Snnn' température cible minimale, attendre pendant la chauffe : 'Rnnn' température cible maximale, attendre jusqu'à refroidissement(Sprinter) : 'Rnnn' température cible précise, attendre tant qu'elle n'est pas précisément atteinte (Marlin) : 'Tn' (optionnel) indique quel extrudeur est concerné ; Exemple : M109 T1 S215 Fixe la température de l'extrudeur 1 à 215°c

M109 dans Marlin, Sprinter (ATmega port) et Duet

Définir la température de l'extrudeur en degrés celcius et attendre qu'elle soit atteinte. Exemple: M109 S185

M110: Set Current Line Number

fived={{yes | teacup=not needed | sprinter=no | marlin=no | repetier=yes | smoothie=yes | bfb=no | machinekit=no }}

Example: M110 N123

Set the current line number to 123. Thus the expected next line after this command will be 124. <br style="clear: both" />

M111: Set Debug Level

fived={{yes | teacup=Debug | sprinter=no | marlin=no | repetier=yes | smoothie=no | reprapfirmware=yes | bfb=no | machinekit=no }}

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:

<Pre> #define DEBUG_ECHO (1«0) #define DEBUG_INFO (1«1) #define DEBUG_ERRORS (1«2) #define DEBUG_DRYRUN (1«3) *repetier-firmware #define DEBUG_COMMUNICATION (1«4)* repetier-firmware </pre>

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.

M112: Arrêt d'urgence

fived={{yes | teacup=yes | sprinter=no | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=yes | bfb=no | machinekit=no }}

Exemple: M112 N'importe quel mouvement en cours est arrêté, et la RepRap s'arrête. Tout les moteurs et le chauffage sont arrêtés. Il peut être redémarrée en appuyant sur le bouton de réinitialisation sur le microcontrôleur maître. voir aussi M0 et M1.

M114: Retourner la postion courante des axes

fived={{yes | teacup=yes | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=yes | bfb=no | machinekit=no }}

Exemple: M114

Cela retourne la position actuelle des axes X, Y, Z et E par rapport à leur 0 respectif de la machine .

Par exemple, la machine retourne une chaîne telle que:

<tt>ok C: X:114.50 Y: 51.70 Z: 140.20 E: 5.40</tt>

La tête d'impression se trouve à 114,50mm en X de son point 0, à 51,70 en Y de son point 0, à 140,20 mm en Z de son point 0, et l'extrudeur est a 5,40mm de matière extruder.

M115: Get Firmware Version and Capabilities

fived={{yes | teacup=yes | sprinter=yes | marlin=yes | repetier=yes | smoothie=no | reprapfirmware=yes | bfb=no | machinekit=no }}

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. See the M408 command for a more comprehensive report on machine capabilities supported by RepRapFirmware. RepRapFirmware-dc42 also uses M115 to tell the firmware about the hardware on which it is running. If the P parameter is present then the integer argument specifies the hardware being used. The following are currently supported: :M115 P0 Automatic board type selection if supported, or default if not :M115 P1 Duet 0.6 :M115 P2 Duet 0.7 :M115 P3 Duet 0.85 ==== M116: Attendre la température ==== fived={{yes | teacup=yes | sprinter=no | marlin=no | repetier=yes | smoothie=no | reprapfirmware=yes | bfb=no | machinekit=no }} Exemple: M116 Attend que `toutes` les températures, ainsi que les variables à variation lente, atteignent les valeurs définies. Voir M109. Le firmware Duet-dc42 (versions 0.78c et supérieures) supporte un paramètre optionnel P, utilisé pour spécifier le numéro d'outil/extrudeur. Si ce paramètre est présent, le système attend uniquement les températures associées à cet outil. C'est utile lors des changements d'extrudeur, pour attendre que le nouvel extrudeur atteigne sa température, sans pour autant attendre que l'extrudeur précédent ait complètement refroidi. Users of Duet-zpl may specify a list of the heaters to be waited for by specifying an 'H' parameter. Duet-zpl v1.08d+ further supports an additional 'C' parameter to wait for the chamber temperature to be reached. ==== M117: Afficher un Message ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=dc42 | bfb=no | machinekit=no }} Exemple: M117 Hello World Cela provoque que le message renseigné soit affiché dans la ligne d'état sur un écran LCD. La commande ci-dessus affichera. ==== M119: Retourner le statut des fin de

courses (endstop) ==== fived={{no | teacup=yes | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=yes | bfb=no | machinekit=no }} Exemple: M119 Retourne l'état actuel X, Y, Z des capteurs de fin de courses. Prends en compte les paramétrages d'inversion du sens de butée des capteurs fin de courses, on peut donc grâce à cette commande savoir si le paramétrage des capteurs de fin de course a été correctement effectué dans le firmware. Si le retour de cette commande indique 'trigered' pour le X-min alors que celui ci n'est pas censé être enclenché indique qu'il faut changer la logique du fin de course dans le firmware. ==== M120: Push ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=yes | reprapfirmware=yes | bfb=no | machinekit=no }} 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 # Current feedrate, and # Whether moves (and separately extrusion) are relative or absolute ==== M121: Pop ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=yes | reprapfirmware=yes | bfb=no | machinekit=no }} Recover the last state pushed onto the stack. ==== M120: Enable endstop detection ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=no | smoothie=no | reprapfirmware=no | bfb=no | machinekit=no }} ==== M121: Disable endstop detection ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=no | smoothie=no | reprapfirmware=no | bfb=no | machinekit=no }} ==== M126: Open Valve ==== fived={{yes | teacup=no | sprinter=no | marlin=yes | repetier=no | smoothie=no | bfb=no | machinekit=no | makerbot=yes }} Example: M126 P500 Open the extruder's valve (if it has one) and wait 500 milliseconds for it to do so. ===== M126 in MakerBot ===== Example: M126 T0 Enables an extra output attached to a specific toolhead (e.g. fan) ==== M127: Close Valve ==== fived={{yes | teacup=no | sprinter=no | marlin=yes | repetier=no | smoothie=no | bfb=no | machinekit=no | makerbot=yes }} Example: M127 P400 Close the extruder's valve (if it has one) and wait 400 milliseconds for it to do so. ===== M127 in MakerBot ===== Example: M127 T0 Disables an extra output attached to a specific toolhead (e.g. fan) ==== M140: Définir la température du plateau chauffant (Rapide) ==== fived={{yes | teacup=yes | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=yes | bfb=no | machinekit=yes }} ; Param-tres : 'Snnn' Température cible ; Exemple : M140 S55 Définit la température du plateau chauffant à 55°C et rend le contrôle à l'hôte immédiatement (`i.e.` avant que la température soit atteinte par le plateau chauffant). Il y a un paramètre 'R' facultatif qui définit la température de veille du plateau chauffant : M140 S65 R40. ==== M141: Set Chamber Temperature (Fast) ==== fived={{no | teacup=1=uses M104 | sprinter=no | marlin=no | repetier=no | smoothie=no | reprapfirmware=zpl, dc42 | bfb=no | machinekit=yes }} 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). Duet-zpl 1.08d+ and Duet-dc42 1.09d+ implement M141 and accept an additional 'H' parameter to set the chamber heater number. ==== M150: Set display color ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=no | smoothie=no | reprapfirmware=no | bfb=no | machinekit=no }} ; Parameters : 'Rnnn' red : 'Unnn' green : 'Bnnn' blue ; Example : M150 R255 U128 B192 Set BlinkM Color via I2C. Range for values: 0-255 ==== M163: Set weight of mixed material ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=yes: 0.92 | smoothie=no | reprapfirmware=no | bfb=no | machinekit=no }} ; Parameters : 'Snnn' extruder number : 'Pnnn' weight Set weight for this mixing extruder drive. ==== M164: Store weights ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=yes: 0.92 | smoothie=no | reprapfirmware=no | bfb=no | machinekit=no }} ; Parameters : 'Snnn' virtual extruder number : 'Pnnn' store to eeprom (P0 = no, P1 = yes) Store weights as virtual extruder S. ==== M190: Wait for bed temperature to reach target temp ==== fived={{no | teacup=no: See M116 | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=yes | bfb=no | machinekit=yes }} ; Parameters : 'Snnn' minimum target temperature, waits until heating : 'Rnnn' accurate target temperature, waits until heating and cooling (Marlin) ; 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. <br style="clear: both" /> ==== M200: Set filament diameter ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=no | bfb=no | machinekit=yes }} Without parameters loads default grid, and with specified extension attempts to load the specified grid. If not available will not modify the current grid. If Z was saved with the grid file, it will load the saved Z with the grid. 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 Optional:_Switch_to_volumetric_E_units and http://wooden-mendel.blogspot.com/2011/09/volumetric-stage-two.html for more information.) M200 D0 or M200 D1.128 ; reset E multiplier to 1, since sqrt(1/pi)*2=1.128 See also 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) Smoothie implements the same thing as Marlin –Arthurwolf (talk) 05:23, 10 November 2014 (PST) ==== M201: Définir l'accélaration maximum d'impréssion ==== fived={{no | teacup=no | sprinter=yes | marlin=yes | repetier=yes | smoothie=no | reprapfirmware=yes | bfb=no | machinekit=no }} Exemple: M201 X1000 Y1000 Z100 E2000 Définit l'accélération que les axes peuvent faire en unités / seconde ^ 2 pour les déplacements d'impression. Par souci de cohérence avec le reste du mouvement G-Code de ce qui devrait être dans les unités / (minute ^ 2), mais qui donne des nombres vraiment incompréhensible et donc on peut se perdre avec tous les zéros. Donc, pour cela, nous utilisons unités / secondes. ==== M202: Définir la vitesse maximum d'accélération de déplacement des axes ==== fived={{no | teacup=no | marlin=yes | repetier=yes | smoothie=no | reprapfirmware=no | bfb=no | machinekit=no }} en unités / s ^ 2 pour des mouvements de déplacement (M202 X1000 Y1000) inutilisés sur Marlin !! ==== M203: Définir la vitesse maximum de déplacement des axes ==== fived={{no | teacup=no | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=yes | bfb=no | machinekit=no }} Exemple: M203 X6000 Y6000 Z300 E10000 Définit la vitesse maximum de déplacement des axes de votre machine quelle peut faire en mm/min (Marlin uses mm/sec). ==== M204: Set default acceleration ==== fived={{no | teacup=no | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=no | bfb=no | machinekit=no }} 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 '`Marlin notes:`' After Mar11-2015, the M204 options have changed in Marlin: P = Printing moves R = Retract only (no X, Y, Z) moves T = Travel (non printing) moves The command "M204 P800 T3000 R9000" set the acceleration for printing movements to 800mm/s^2, for travels to 3000mm/s^2 and for retracts to 9000mm/s^2. ===== M204 Repetier ===== M204 X[Kp] Y[Ki] Z[Kd] - Set PID parameter. Values are 100*real value. ==== M205: Advanced settings ==== fived={{no | teacup=no | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | bfb=no | machinekit=no }} 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 ===== M205 Repetier ===== Output EEPROM settings. ==== M206: ==== fived={{no | teacup=no | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | bfb=no | machinekit=no }} ===== M206 Marlin, Sprinter, Smoothie - 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. With Marlin 1.0.0 RC2 a negative value for z lifts(!) your printhead. ===== M206 Repetier - Set eeprom value ===== M206 T[type] P[pos] [Sint(long] [Xfloat] Set eeprom value Example: M206 T3 P39 X19.9 Set Jerk to 19.9 ==== M207: Set retract length ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=no | smoothie=yes | reprapfirmware=no | bfb=no |*

machinekit=no}} ; Parameters : 'Snnn' positive length to retract, in mm : 'Fnnn' feedrate, in mm/min : 'Znnn' additional zlift/hop ; Example : M207 S4.0 F2400 Z0.075 Sets retract length, stays in mm regardless of M200 setting ==== M208: Set axis max travel ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=no | reprapfirmware=yes | bfb=no | machinekit=no}} Example: M208 X250 Y210 Z180 The values specified set the software limits for axis travel in the positive direction. RepRapPro's version of Marlin uses M208 this way. Send M503 to see the current values. The value can be saved to EEPROM using the M500 command. With Duet-dc42 firmware, on a Cartesian printer 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. ==== M208: Set unretract length ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=no | smoothie=yes | reprapfirmware=no | bfb=no | machinekit=no }} ; Parameters : 'Snnn' positive length surplus to the M207 Snnn, in mm : 'Fnnn' feedrate, in mm/sec Sets recover=unretract length. ==== M209: Enable automatic retract ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=yes | smoothie=no | reprapfirmware=no | bfb=no | machinekit=no }} 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 ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=no | bfb=no | machinekit=no }} Example: M210 X1000 Y1500 Set the feedrates used for homing to the values specified in mm per minute. ==== M211: Disable/Enable software endstops ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=no | bfb=no | machinekit=no }} The boolean value S 1=enable or 0=disable controls state of software endstop. The boolean value X, Y or Z 1=max endstop or 0=min endstop selects which endstop is controlled. Example: M211 X1 Y1 Z1 S0 Disables X,Y,Z max endstops Example: M211 X0 S1 Enables X min endstop Example: M211 Prints current state of software endstops. ==== M212: Set Bed Level Sensor Offset ==== fived={{no | teacup=no | sprinter=no | marlin=yes* | repetier=no | smoothie=no | bfb=no | machinekit=no }} This G-Code command is known to be available in the newer versions of PrintrBot's branch of Marlin. It may not be available in other firmware. Example: M212 Z-0.2 Set the Z home to 0.2 mm lower than where the sensor says Z home is. This is extremely useful when working with printers with hard-to-move sensors, like the PrintrBot Metal Plus. PrintrBot suggests that the user make minor (0.1-0.2) adjustments between attempts and immediately executes M500 & M501 after setting this. ==== M218: Set Hotend Offset ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=no | smoothie=no | reprapfirmware=no | bfb=no | machinekit=no }} Sets hotend offset (in mm): T<extruder_number> X<offset_on_X> Y<offset_on_Y>. Example: M218 T1 X50 Y0.5 ==== M231: Set OPS parameter ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=yes | smoothie=no | bfb=no | machinekit=no }} M231 S[OPS_MODE] X[Min_Distance] Y[Retract] Z[Backslash] F[ReatrctMove] ==== M232: Read and reset max. advance values ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=yes | smoothie=no | bfb=no | machinekit=no }} ==== M240: Déclencher l'appareil photo ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=no | smoothie=no | bfb=no | machinekit=no }} Example: M240 Déclenche un appareil photo connecté sur la carte. ==== M250: Set LCD contrast ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=no | smoothie=no | bfb=no | machinekit=no }} Example: M250 C20 Sets LCD contrast C<contrast value> (value 0..63), if available. ==== M251: Measure Z steps from homing stop (Delta printers) ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=yes | smoothie=no | bfb=no | machinekit=no }} M251 S0 - Reset, S1 - Print, S2 - Store to Z length (also EEPROM if enabled) (This is a Repetier-Firmware only feature) ==== M280: Définir la postion du servomoteur ==== fived={{no | teacup=no | sprinter=no | marlin=yes | smoothie=no | smoothie=no | bfb=no | machinekit=yes }} (Marlin, Repetier M340) Définir la postion absolue du servomoteur. P: determine le servomoteur , S: Angle ou Microsecondes (Marlin) ==== M300: Jouer un

son ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=yes | smoothie=no | reprapfirmware=dc42 | bfb=no | machinekit=yes }} ; Parametres : 'Snnn' frequence en Hz : 'Pnnn' durer en millisecondes ; Exemple : M300 S300 P1000 Jouer un bip sonore de 300Hz pendant 1000 millisecondes, utiliser le pour notifier des événements importants comme la fin de l'impression. ==== M301: Définir les paramétrage du PID ==== fived={{no | teacup=no: See M13[0-3]) | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware= yes | bfb=no | machinekit=no }} ; Parametere : 'Pnnn' proportionel (Kp) : 'Innn' intégral (Ki) : 'Dnnn' derivé (Kd) ; Exemple : M301 P1 I2 D3 ; Marlin Définir la valeur de la hotend pour la Proportionel (P), l'Intégral (I) et la Derivé (D). Voir PID Tuning. ==== M302: Autoriser l'extrusion à froid ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=yes: 0.92 | smoothie=no | reprapfirmware= yes | bfb=no | machinekit=no }} Ceci indique à l'imprimante de permettre un mouvement du moteur de l'extrudeur meme si la hotend n'est pas à la température d'impression Exemple: M302 ==== M303: Executer un paramétrage automatic du PID ==== fived={{no | teacup=no | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=no | bfb=no | machinekit=no }} PID Tuning se réfère à un algorithme de contrôle utilisé dans certains repraps pour les hotends et les lits chauffants. Cette commande génère le proportionnel (Kp), intégral (Ki), et (Kd) la valeur des produits dérivés pour la hotend ou lit (E-1). Envoyer le code M303 et attender la fin du programme. Pid de la hotend: M303 E0 S<température> C< nombre de cycles> Exemple: M303 E0 C8 S175 le firmware va exécuter un test de la hotend 1 de 8 cycles avec une consigne de 175°. ==== M304: Définir les parametre PID du Bed (plateau chauffant) ==== fived={{no | teacup=no | sprinter=no | marlin=yes | smoothie=no: See M301 | reprapfirmware=dc42 | bfb=no | machinekit=no }} ; Parameters : 'Pnnn' proportionnelle (Kp) : 'Innn' intégrale (Ki) : 'Dnnn' dérivé (Kd) ; Exemples : M304 P1 I2 D3 ; set kP=3, kI=2, kD=3 : M301 P1 I2 D3 T0.7 H0 B20 W127 ; Duet-dc42 firmware : M304 ; Retourner les parametres Définit les valeurs proportionnelle, intégrale et dérivé pour le bed (plateau chauffant). Regarder aussi PID Tuning. ==== M305: Set thermistor and ADC parameters ==== fived={{no | teacup=no | sprinter=no | marlin=no | smoothie=yes | reprapfirmware=yes | bfb=no | machinekit=no }} Sets the parameters for temperature measurement. Supported by RepRapFirmware from 0.78c, and Duet-dc42 firmware. Example: M305 P1 T100000 R1000 B4200 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. All parameters other than P are optional. If only the P parameter is given, the existing values are displayed. Additionally, Duet-dc42 firmware supports an ADC correction functionality and a thermistor selection facility. Example: M305 P1 T100000 R1000 B4200 H14 L-11 X2 Here the ADC high end correction (H parameter) is 14, the ADC low end correction (L parameter) is -11, and thermistor input #2 is used to measure the temperature of heater #1. ==== M306: set home offset calculated from toolhead position ==== fived={{no | teacup=no | sprinter=no | marlin=no | smoothie=yes | bfb=no | machinekit=no }} Example: M306 Z0 The values specified are added to the calculated end stop position when the axes are referenced. The calculated value is derived from the distance of the toolhead from the current axis zero point. The user would typically place the toolhead at the zero point of the axis and issue the M306 command. This value can be saved to EEPROM using the M500 command (as M206 value). Implemented in Smoothieware ==== M351: Toggle MS1 MS2 pins directly ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=no | smoothie=no | bfb=no | machinekit=no }} Example: M351 ==== M355: Turn case lights on/off ==== fived={{no | teacup=use M106 | sprinter=no | marlin=no | repetier=yes: 0.92.2 | smoothie=no | reprapfirmware=no | bfb=no | machinekit=no }} ; Examples : M355 S1 ; Enable lights : M355 S0 ; Disable lights : M355 ; Report status Every call or change over LCD menu sends a state change for connected hosting software like <pre> Case lights on Case lights off No case lights </pre> ==== M360: Report firmware configuration ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=yes: 0.92.2 | smoothie=no | reprapfirmware=no | bfb=no | machinekit=no }} ; Target This command helps hosting software to detect configuration details, which the user would need to enter otherwise. It should reduce configuration time considerably if

supported. ; Example : M360 ;Response: <pre> Config:Baudrate:250000 Config:InputBuffer:127 Config:NumExtruder:2 Config:MixingExtruder:0 Config:HeatedBed:0 Config:SDCard:1 Config:Fan:1 Config:LCD:1 Config:SoftwarePowerSwitch:1 Config:XHomeDir:-1 Config:YHomeDir:-1 Config:ZHomeDir:-1 Config:SupportG10G11:1 Config:SupportLocalFilamentchange:1 Config:CaseLights:0 Config:ZProbe:1 Config:Autolevel:0 Config:EEPROM:1 Config:PrintlineCache:24 Config:JerkXY:30.00 Config:JerkZ:0.30 Config:RetractionLength:3.00 Config:RetractionLongLength:13.00 Config:RetractionSpeed:40.00 Config:RetractionZLift:0.00 Config:RetractionUndoExtraLength:0.00 Config:RetractionUndoExtraLongLength:0.00 Config:RetractionUndoSpeed:0.00 Config:XMin:0.00 Config:YMin:0.00 Config:ZMin:0.00 Config:XMax:250.00 Config:YMax:150.00 Config:ZMax:90.00 Config:XSize:250.00 Config:YSize:150.00 Config:ZSize:90.00 Config:XPrintAccel:250.00 Config:YPrintAccel:250.00 Config:ZPrintAccel:100.00 Config:XTravelAccel:250.00 Config:YTravelAccel:250.00 Config:ZTravelAccel:100.00 Config:PrinterType:Cartesian Config:MaxBedTemp:120 Config:Extr.1:Jerk:50.00 Config:Extr.1:MaxSpeed:100.00 Config:Extr.1:Acceleration:10000.00 Config:Extr.1:Diameter:0.00 Config:Extr.1:MaxTemp:220 Config:Extr.2:Jerk:50.00 Config:Extr.2:MaxSpeed:100.00 Config:Extr.2:Acceleration:10000.00 Config:Extr.2:Diameter:0.00 Config:Extr.2:MaxTemp:220 </pre> ==== SCARA calibration codes (Morgan) ==== In order to ease calibration of Reprap Morgan, the following M-codes are used to set the machine up fived={{no | teacup=no | sprinter=no | marlin=partial | repetier=no | smoothie=yes | reprapfirmware=no | bfb=no | machinekit=no }} ==== M360: Move to Theta 0 degree position ==== fived={{no | teacup=no | sprinter=no | marlin=experimental | repetier=no | smoothie=yes | reprapfirmware=no | bfb=no | machinekit=no }} 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. Smoothieware: M360 P0 will take the current position as parallel to the platform edge, and store the offset in the homing trim offset (M666) No further user interaction is needed. ==== M361: Move to Theta 90 degree position ==== fived={{no | teacup=no | sprinter=no | marlin=experimental | repetier=no | smoothie=yes | reprapfirmware=no | bfb=no | machinekit=no }} 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. Smoothieware: M360 P0 will accept the current position as 90deg to platform edge. New steps per angle is calculated and entered into memory (M92) No further user interaction is required, except to redo M360. ==== M362: Move to Psi 0 degree position ==== fived={{no | teacup=no | sprinter=no | marlin=experimental | repetier=no | smoothie=yes | reprapfirmware=no | bfb=no | machinekit=no }} Arms move to Psi 0 degree. Check only after other Theta calibrations ==== M363: Move to Psi 90 degree position ==== fived={{no | teacup=no | sprinter=no | marlin=experimental | repetier=no | smoothie=yes | reprapfirmware=no | bfb=no | machinekit=no }} Arms move to Psi 90 degree. Check only after other Theta calibrations ==== M364: Move to Psi + Theta 90 degree position ==== fived={{no | teacup=no | sprinter=no | marlin=experimental | repetier=no | smoothie=yes | reprapfirmware=no | bfb=no | machinekit=no }} 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. Smoothieware: M364 P0 will accept the current position as 90deg between arms. The offset is stored as a trim offset (M666) and no further user interaction is required except to save all changes via M500 ==== M365: SCARA scaling factor ==== fived={{no | teacup=no | sprinter=no | marlin=experimental | repetier=no | smoothie=yes | reprapfirmware=no | bfb=no | machinekit=no }} Adjust X Y and Z scaling by entering the factor. 100% scaling (default) is represented by 1 ==== M366: SCARA convert trim ====

fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=yes | reprapfirmware=no | bfb=no | machinekit=no }} Executing this command translates the calculated trim values of the SCARA calibration to real home offsets. This prevents the home and trim movement after calibration. ==== M370: Morgan manual bed level - clear map ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=yes | reprapfirmware=no | bfb=no | machinekit=no }} 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 ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=yes | reprapfirmware=no | bfb=no | machinekit=no }} 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 ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=yes | reprapfirmware=no | bfb=no | machinekit=no }} 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 ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=yes | reprapfirmware=no | bfb=no | machinekit=no }} End calibration mode and enable z correction matrix. Does not save current matrix ==== M374: Save calibration grid ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=yes | reprapfirmware=no | bfb=no | machinekit=no }} Saves the calibration grid. (Smoothieware) ; Usage : M374 : M374 <file extension> Z Without parameters safes the grid into the default grid file that gets loaded at boot Parameter specifies the extension of the grid file - useful for special grid files such as for a special print surface like a removable print plate. Addition of Z will additionally save the M206 Z homing offset into the grid file ==== M375: Display matrix / Load Matrix ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=yes | reprapfirmware=no | bfb=no | machinekit=no }} Display the bed level calibration matrix (Marlin) Load Grid matrix file (Smoothieware) ; Usage : M375 : M375 <file extension> Without parameters loads default grid, and with specified extension attempts to load the specified grid. If not available will not modify the current grid. If Z was saved with the grid file, it will load the saved Z with the grid. ==== M380: Activate solenoid ==== fived={{no | teacup=no | sprinter=no | marlin=yes | smoothie=no | bfb=no | machinekit=no }} Example: M380 Activates solenoid on active extruder. ==== M381: Disable all solenoids ==== fived={{no | teacup=no | sprinter=no | marlin=yes | smoothie=no | bfb=no | machinekit=no }} Example: M381 ==== M400: Wait for current moves to finish ==== fived={{no | teacup=no | sprinter=yes | marlin=yes | repetier=yes | smoothie=yes | bfb=no | machinekit=yes }} Finishes all current moves and and thus clears the buffer. That's identical to <code>G4 P0</code>. Example: M400 ==== M401: Lower z-probe ==== fived={{no | teacup=no | sprinter=no | marlin=yes | smoothie=no | bfb=no | machinekit=no }} Example: M401 Lower z-probe if present. ==== M402: Raise z-probe ==== fived={{no | teacup=no | sprinter=no | marlin=yes | smoothie=no | bfb=no | machinekit=no }} Example: M402 Raise z-probe if present. ==== M404: Filament width and nozzle diameter ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=no | smoothie=no | reprapfirmware=partial | bfb=no | machinekit=no }} Example: M404 N1.75 N<dia in mm> Enter the nominal filament width (3mm, 1.75mm) or will display nominal filament width without parameters. While Marlin only accepts the N parameter, Duet-zpl further allows to specify the nozzle diameter (in mm) via the D-parameter. It is used to properly detect the first layer height when a new print is being started. Example: M404 N3.0 D1.0 ==== M405: Allumage du capteur de filament ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=no | smoothie=no | reprapfirmware=no | bfb=no | machinekit=no }} Exemple: M405 Allume le contrôle d'extrusion du filament. D optionnel <Retard en cm> pour définir le délai en centimètres entre le capteur et l'extrudeur. ==== M406: Extinction du capteur de filament ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=no | smoothie=no | bfb=no | machinekit=no }} Exemple: M406 Extinction du capteur de filament. ==== M407: Display filament diameter ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=no | smoothie=no | bfb=no | machinekit=no | reprapfirmware=partial }} Example: M407

Displays measured filament diameter. In RepRapFirmware M407 does the same as M404. ==== M421: Set a Mesh Bed Leveling Z coordinate ==== M421 - Set a single Z coordinate in the Mesh Leveling grid. X<index> Y<index> Z<offset in mm> ==== M500: Défini les paramètres dans l'EEPROM ==== fived={{no | teacup=no | sprinter=yes | marlin=yes | repetier=yes | reprapfirmware=dc42, zpl | bfb=no | machinekit=no }} Exemple: M500 Sauvegarder les paramètres actuels dans l'EEPROM. ==== M501: Lire les parametres à partir de l'EEPROM ==== fived={{no | teacup=no | sprinter=yes | marlin=yes | repetier=yes | reprapfirmware=dc42, zpl | bfb=no | machinekit=no }} Exemple: M501 Lit les paramètres enregistrés dans l'EEPROM. ==== M502: Réinitialiser "réglages d'usine." ==== fived={{no | teacup=no | sprinter=yes | marlin=yes | repetier=yes | bfb=no | machinekit=no }} Example: M502 Cette commande réinitialise tous les paramètres ajustables à leurs valeurs par défaut, comme indiqué dans le firmware. Cela ne réinitialise pas les paramètres stockés dans la mémoire EEPROM, donc il doit être suivi avec M501 si vous voulez le faire. ==== M503: Print settings ==== fived={{no | teacup=no | sprinter=yes | marlin=yes | reprapfirmware=dc42, zpl | bfb=no | machinekit=no }} Example: M503 This command asks the firmware to reply with the current print settings stored in EEPROM. The reply output includes the G-Code commands to produce each setting. For example, the Steps Per Unit values are displayed as an M92 command. ==== M540: Enable/Disable "Stop SD Print on Endstop Hit" ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=no | smoothie=no | reprapfirmware=no | bfb=no | machinekit=no }} ; Parameters : 'Snnn' state, S1=enable, S0=disable ; Example : M540 S1 ==== M557: Set Z probe point ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=yes | reprapfirmware=yes | bfb=no | machinekit=no }} 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 ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=no | reprapfirmware=yes | bfb=no | machinekit=no }} Example: M558 P0 X1 Y0 Z1 A Z probe may be a switch, an IR proximity sensor, or some other device. This selects which to use. P0 indicates that no Z probe is present. 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. P4 selects a switch (on the Duet, this must be connected to the E0 endstop pins). The X, Y and Z parameters specify whether each axis uses the Z probe for homing or not. If the parameter is nonzero, the Z probe is used for homing that axis. if the parameter is zero, the endstop switch for that axis is used for homing instead. See also G31 and G32. Duet-dc42 and zpl-dc42 firmware support an additional R parameter, which specifies the modulation channel. Channel 0 (the default) uses the standard Z probe modulation pin on the Duet 0.6. Channel 1 selects the alternative Z probe modulation pin on the Duet 0.7. Note that on RADDS electronics, the R parameter is accepted but ignored: the selected channel is reported, but the same RADDS pin is always used. Duet-dc42 firmware versions 1.00e onwards supports additional parameter H. This specifies the dive height (default 3mm) from which probing is done in response to a G30 command when the P parameter is present, or a G32 command. ==== M559: Upload configuration file ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=no | reprapfirmware=yes | bfb=no | machinekit=no }} 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 ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=no |

*reprapfirmware=yes | bfb=no | machinekit=no }} 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 <pre><!– **EoF** –></pre>. 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 ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=yes | reprapfirmware=yes | bfb=no | machinekit=no }} 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 ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=no | reprapfirmware=yes | bfb=no | machinekit=no }} 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 or remove a tool ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=no | reprapfirmware=yes | bfb=no | machinekit=no }} Example: M563 P3 D0:5:6 H1:3 Tools are usually (though not necessarily) extruders. The P field specifies the tool number. Tool numbers can have any positive integer value and 0. 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: <pre> G1 X90.6 Y13.8 E2.24:2.24:15.89 G1 X70.6 E0:0:42.4 </pre> 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. If you use the M563 command with a P value for a tool that has already been defined, that tool is redefined using the new values you provide. Duet-dc42 firmware supports an additional form of the M563 command. The command: <pre> M563 S1 </pre> means add 1 (the value of the S parameter) to all tool numbers found in the remainder of the current input stream (e.g. the current file if the command is read from a file on the SD card), or until a new M563 command of this form is executed. The purpose of this is to provide compatibility between the Duet firmware, in which tool numbers typically start at 1, and programs such as slic3r that assume tools are numbered from zero. Duet-zpl firmware allows the deletion of existing tools if M563 is called in this way: <pre> M563 P1 D-1 H-1 </pre> ==== M564: Limit axes ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=no | reprapfirmware=yes | bfb=no | machinekit=no }} 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 ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=yes | reprapfirmware= no: See G31 | bfb=no | machinekit=no }} 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. Example: M572 P3 S0.06 This sets the pre-compensation time in seconds (S parameter) for Bowden extruder elasticity for the specified drive (P parameter). Supported by RepRapFirmware-dc42. Normally, compensation should be applied to extruder drives only (drives 3 and higher). Pre-compensation causes the extruder drive position to be increased by an*

additional amount proportional to the rate of extrusion. At the end of a move when the extrusion rate is decreasing, this may result in the extruder drive moving backwards (i.e. retracting). Therefore, if you enable this feature, you may need to reduce the amount of retraction you use in your slicing program to avoid over-retraction. ==== M573: Report heater PWM ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=no | reprapfirmware=yes | bfb=no | machinekit=no }} Example: M573 P1 This gives a running average (usually taken over about five seconds) of the PWM to the heater specified by the P field. If you know the voltage of the supply and the resistance of the heater this allows you to work out the power going to the heater. ==== M574: Set endstop configuration ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=no | reprapfirmware=dc42 | bfb=no | machinekit=no }} Example: M574 X1 Y2 Z0 S1 This defines the type of endstop switch or opto sensor that the printer has for each axis: 0 = none, 1 = low end, 2 = high end. The optional S parameter defines whether the endstop input is active high (S1, the default) or low (S0). Intended for use with boards that provide a single endstop input for each axis that may be used for either a high or a low end endstop, such as the Duet. Supported by Duet-dc42 firmware. On delta printers, the XYZ parameters refer to the towers and the endstops should normally all be high end. ==== M575: Set serial comms parameters ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=no | reprapfirmware=dc42 | bfb=no | machinekit=no }} Example: M575 P1 B57600 S1 This sets the communications parameters of the serial comms channel specified by the P parameter. P0 specifies the main serial interface (typically a USB port, or serial-over-USB), while P1 specifies an auxiliary serial port (for example, the port used to connect a PanelDue). The B parameter is the required baud rate (this parameter is typically ignored if the port is a true USB port). The S parameter is a bitmap of features. The lowest bit, if set, specifies that only commands that include a valid checksum should be accepted from this comms channel. ==== M576: Set axis/extruder drive mapping ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=no | reprapfirmware=not yet | bfb=no | machinekit=no | makerbot=no }} Example: M576 X0 Y1 Z2:3 A4 E5:6:7 This maps each letter to the specified drive number(s). In this example the Z axis configuration consists of two motors that need to be controlled simultaneously. An extra axis 'A' is also defined for custom purposes (rotary nozzle, cutter tool etc.) while the remaining axes are mapped to extruder drives. See also M569, which also provides for setting the drive mapping. ==== M577: Wait until endstop is triggered ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=no | reprapfirmware=Duet-zpl | bfb=no | machinekit=no | makerbot=no }} Example: M577 E0 S1 Wait for an endstop switch to be pressed. The example above will wait until the first extruder endstop is triggered. The following trigger types may be used using the 'S' parameter: 0: Endstop not hit 1: Low endstop hit 2: High endstop hit 3: Near endstop (only Z probe) ==== M578: Fire inkjet bits ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=no | reprapfirmware=yes | bfb=no | machinekit=no | makerbot=no }} Example: M578 P3 S5 This fires inkjet head 3 (the P field) using the bit pattern specified by the S field. The example shown would fire bits 101. If the P parameter is ommitted inkjet 0 is assumed. This is a version of the M700 command used by the Inkshield, but unfortunately M700 is already taken so cannot be used for that in the standard. ==== M579: Scale Cartesian axes ==== Example: M579 X1.0 Y0.997 Z1.001 On a Cartesian RepRap you can get prints exactly the right size by tweaking the axis steps/mm using the M92 G Code above. But this does not work so easily for Delta and other RepRaps for which there is cross-talk between the axes. This command allows you to adjust the X, Y, and Z axis scales directly. So, if you print a part for which the Y length should be 100mm and measure it and find that it is 100.3mm long then you set Y0.997 (= 100/100.3). ==== M580: Select Roland ==== Example: M580 R1 PVS4;!VZ2;!MC1; This is not really anything to do with RepRap, but it is convenient. The [http://www.rolanddg.com/product/3d/3d/mdx-20_15/mdx-20_15.html little Roland mills] are very widely available in hackerspaces and maker groups, but annoyingly they don't speak G Codes. As all

RepRap firmware includes a G-Code interpreter, it is often easy to add functions to convert G Codes to [http://altlab.org/d/content/m/pangelo/ideas/rml_command_guide_en_v100.pdf Roland RML language]. M580 selects a Roland device for output if the R field is 1, and returns to native mode if the R field is 0. The optional P string is sent to the Roland if R is 1. It is permissible to call this repeatedly with R set to 1 and different strings in the P field to communicate directly with a Roland. ==== M600: Set line cross section ==== fived={{no | teacup=no | sprinter=no | marlin=no | bfb=no | machinekit=yes }} Example: M600 P0.061 Sets the cross section for a line to extrude in velocity extrusion mode. When the extruder is enabled and movement is executed the amount of extruded filament will be calculated to match the specified line cross section. ==== M600: Filament change pause ==== fived={{no | teacup=no | sprinter=no | marlin=yes | reprapfirmware=no | bfb=no | machinekit=no }} Example: M600 Pause for filament change X[pos] Y[pos] Z[relative lift] E[initial retract] L[later retract distance for removal]. ==== M605: Set dual x-carriage movement mode ==== fived={{no | teacup=no | sprinter=no | marlin=yes | smoothie=no | bfb=no | machinekit=no }} Example: M605 S1 Sets the dual x-carriage movement mode: S<mode> [X<duplication x-offset> R<duplication temp offset>]. M605 S0: Full control mode. The slicer has full control over x-carriage movement M605 S1: Auto-park mode. The inactive head will auto park/unpark without slicer involvement M605 S2 [Xnnn] [Rmmm]: Duplication mode. The second extruder will duplicate the first with nnn millimeters x-offset and an optional differential hotend temperature of mmm degrees. E.g., with “M605 S2 X100 R2” the second extruder will duplicate the first with a spacing of 100mm in the x direction and 2 degrees hotter. ==== M665: Set delta configuration ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=no | smoothie=yes | reprapfirmware=dc42 | bfb=no | machinekit=no }} Example: M665 L250 R160 S200 (Marlin), M665 L250 R160 B80 H240 X0 Y0 Z0 (RepRapFirmware-dc42) Set the delta calibration variables. L = diagonal rod length, R = delta radius. For Marlin: S = segments per second. For RepRapFirmware-dc42: B = safe printing radius, H = nozzle height above bed when homed after allowing for the endstop corrections; X, Y, Z = X, Y and Z tower angular offsets from the ideal (i.e.equilateral triangle) positions, in degrees, measured anti-clockwise looking down on the printer. : I don't think it's a good idea to have two different implementations for the same G-code, and I also question the practical value of specifying the print bed radius when defining a delta configuration, since many delta printers use a square or rectangular print bed. So perhaps we should stick to the Marlin-defined command as the definition for this command, and use a different command or set of commands to define print bed shape and size. –AndrewBCN (talk) 23:10, 31 January 2015 (PST) : The implementations are not different, they have the same L and R parameters, but each has additional parameters that are not relevant to the other implementation. I'm not against defining a new Gcode to define bed size and shape; however, you can already define the limits of a rectangular print area using M208. Even when a delta printer has a square bed, the printable area is not square. It is usually taken to be circular, although it is in reality more complicated. My purpose in adding the B parameter was to make it easy to define a radius outside which movement will not normally be attempted. I have changed “bed radius” to “safe printing radius” in the text to help clarify this. –dc42 ==== M666: Définir l'ajustement des endstops des Delta ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=no | smoothie=yes | reprapfirmware=dc42 | bfb=no | machinekit=no }} Exemple M666 X-0.1 Y+0.2 Z0 Définir l'ajustement des endstops des Delta. ==== M667: Select CoreXY mode ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=no | reprapfirmware=dc42 | bfb=no | machinekit=no }} M667 S0 selects Cartesian mode (unless the printer is configured as a delta using the M665 command). Forward motion of the X motor moves the head in the +X direction. Similarly for the Y motor and Y axis, and the Z motor and Z axis. This is the default state of the firmware on power up. M667 S1 selects CoreXY mode. Forward movement of the X motor moves the head in the +X and +Y directions. Forward movement of the Y motor moves the head in the -X and +Y directions. M667 S2 selects CoreXZ mode. Forward movement of the X motor moves the head in the +X and +Z directions. Forward movement of the Z motor moves the head in the -X and +Z directions. M667 S3 selects CoreYZ mode. Forward movement of the Y motor moves the head in the +Y and +Z directions. Forward movement of the Z motor moves the head in the -Y

*and +Z directions. Additional parameters X, Y and Z may be given to specify factors to scale the motor movements by for the corresponding axes. For example, to specify a CoreXZ machine in which the Z axis moves 1/3 of the distance of the X axis for the same motor movement, use M667 S2 Z3. The default scaling factor after power up is 1.0 for all axes. To change the motor directions, see the M569 command. ==== M700: Level plate ==== fived={{no | teacup=no | sprinter=no | marlin=bq | repetier=no | smoothie=no | bfb=no | machinekit=no }} [1] only in bq-Marlin Firmware Example: M700 Script to adjust the plate level. ==== M701: Load filament ==== fived={{no | teacup=no | sprinter=no | marlin=bq | repetier=no | smoothie=no | bfb=no | machinekit=no }} [1] only in bq-Marlin Firmware Example: M701 ==== M702: Unload filament==== fived={{no | teacup=no | sprinter=no | marlin=bq | repetier=no | smoothie=no | bfb=no | machinekit=no }} [1] only in bq-Marlin Firmware Example: M702 ==== M851: Set Z-Probe Offset ==== marlin={{yes }} Sets the Z-Probe Offset saved in the EEPROM and this setting also works like M206 as well and this has priority over the z probe offset you set in marlin configuration.h setting Example: M851 Z-4 The example above will set the z-probe offset EEPROM setting to -4mm below the nozzle and enables the nozzle travel 4mm lower than the probe triggered position. It is however, a good idea to keep the setting inside your configuration.h as well for your own future reference. This command appears on pronterface after marlin dev 1.1.0, it is unknown that this command can be used on marlin 1.0.0 ==== M906: Set motor currents ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=M907? | reprapfirmware=yes | bfb=no | machinekit=no | redeem=yes }} Example: M906 X300 Y500 Z200 E350 Sets the currents to send to the stepper motors for each axis. The values are in milliamps. The dc42 fork of RepRapFirmware supports an additional I parameter. This is the percentage of normal that the motor currents should be reduced to when the printer becomes idle but the motors have not been switched off. ==== M907: Set digital trimpot motor ==== fived={{no | teacup=no | sprinter=no | marlin=yes | smoothie=yes | repetier=no | bfb=no | machinekit=no | redeem=yes }} Set digital trimpot motor current using axis codes (X, Y, Z, E, B, S). In [https://bitbucket.org/intelligentagent/redeem/src/6153607ded91c100fb4e41e936e6d045e19eda29/redeem/gcodes/M907.py?at=slave_stepper Redeem], it sets the current in A (where M906 does in mA). ==== M908: Control digital trimpot directly ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=yes: 0.92 | smoothie=no | bfb=no | machinekit=no }} M908 P<pin> S<current> ==== M909: Set microstepping ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=no | bfb=no | machinekit=no | redeem=yes }} Example: M909 X3 Y5 Z2 E3 Set the microstepping value for each of the steppers. In Redeem this is implemented as 2^value, so
 M909 X2 sets microstepping on X-axis to 2^2 = 4,
 M909 Y3 sets microstepping on Y-axis to 2^3 = 8 etc. ==== M910: Set decay mode ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=no | bfb=no | machinekit=no | redeem=yes }} Example: M910 X3 Y5 Z2 E3 Set the decay mode for each stepper controller The decay mode controls how the current is reduced and recycled by the H-bridge in the stepper motor controller. It varies how the implementations are done in silicone between controllers. Typically you have an on phase where the current flows in the target current, then an off phase where the current is reversed and then a slow decay phase where the current is recycled. ==== M928: Start SD logging ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=no | smoothie=no | bfb=no | machinekit=no }} Example: M928 filename.g Ended by M29. ==== M998: Request resend of line ==== fived={{no | teacup=no | sprinter=no | marlin=no | repetier=no | smoothie=no | bfb=no | machinekit=no | reprapfirmware=yes }} Example: M998 P34 Request a resend of line 34. In some implementations the input-handling code overwrites the incoming 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: Redémarrez après avoir été arrêté par erreur ==== fived={{no | teacup=no | sprinter=no | marlin=yes | repetier=no | smoothie=yes | reprapfirmware=yes | bfb=no | machinekit=no }} Exemple: M999 Redémarre le firmware apres une erreur. == Other commands ==*

*==== G: List all G-codes ==== reprapfirmware={{no }} Example: G This lists all implemeted G-codes in the firmware with description and sends it back to the host.
 (Note: this has been implemented in Redeem, and so is only a proposition) ==== M: List all M-codes ==== Example: M This lists all implemeted M-codes in the firmware with description and sends it back to the host.
 (Note: this has been implemented in Redeem, and so is only a proposition) ==== T: Select Tool ==== fived={{yes | teacup=yes | sprinter=no | marlin=yes | repetier=yes | smoothie=yes | reprapfirmware=yes }} Example: T1 Select tool (or in older implementations extruder) number 1 to build with. The sequence followed is: # Set the current tool to its standby temperatures specified by G10 (see above), # Set the new tool to its operating temperatures specified by G10 and wait for `'all'` temperatures to stabilise, # Apply any X, Y, Z offset for the new tool specified by G10, # 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: # Actions to do with the old tool before it is released - macro name: `'tfreeN.g'` where N is the tool number; # (Old tool is released); # Actions to do with the new tool before it is selected - macro name: `'tpreN.g'` where N is the tool number; # (New tool is selected); and # 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 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 codes M500-M503 to manipulate EEPROM values. Sprinter has implemented the following commands to manipulate EEPROM [https://github.com/kliment/Sprinter/commit/4b1b0f1d96d2be2ed3941095f40a5c2d2bbb943d Commit message]. Teacup uses codes M130-M136 to set, read, and save some parameters. == 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°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 ''. On the latest version of Pronterface and soon Octoprint a special comment of the form: ' action:command' is allowed to be sent from the firmware, the command can currently be pause, resume or disconnect which will execute those commands on the host. As this is also a comment other hosts will just ignore these commands. 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': 1. 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? 2. The dc42 fork of RepRapFirmware responds to some commands with a reply string in JSON format, terminated by a newline. This allows later firmware revisions to include additional information without confusing clients (e.g. PanelDue) that do not expect it, and to make responses self-describing so that the client will not be confused if responses are delayed or lost. The commands affected are: * M105 S2 * M105 S3 * M20 S2 * M36 * M408 == Proposal for sending multiple lines of G-code == So far, this is a proposal, open for discussion. ==== Problem to solve ==== When using Marlin firmware or emulating Marlin, 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 communications up. This slows down communication and limits the number of commands that can be sent per second to the printer controller, as the USB stack on the host and the serial interface driver on the Arduino add their own latencies (up to 10 milliseconds). This is not a problem for other controller electronics using native USB such as the Duet, because the standard serial-over-USB drivers provide flow control, so the host software can be configured so as not to wait for the 'ok'. For more details on this proposal, some suggested solutions and comments, please see GCODE_buffer_multiline_proposal == Alternatives to G-code == : Main article: alternatives to G-code Several people have suggested using STEP-NC or some other control language; or perhaps designing a completely new control language. G-code/fr Firmware/fr Software/fr*