LinuxCÑC

LinuxCNC en castellano

NGCGUI

linuxcnces 12 noviembre, 2019 Doc. oficial Editar

Documentación Original de LinuxCNC

Indice de contenido

1. Información general
2. Configuraciones de demostración
3. Ubicaciones de la biblioteca
4. Uso independiente
4.1. NGCGUI independiente
4.2. PYNGCGUI independiente
5. Incrustar NGCGUI
5.1. Incrustar NGCGUI en Axis
5.2. Incrustar PYNGCGUI como una pestaña gladevcp en una gui
5.3. Elementos adicionales del archivo INI necesarios para ngcgui o pyngcgui
5.4. Truetype Tracer
5.5. Especificaciones de ruta del archivo INI
5.6. Resumen de los detalles del elemento del archivo INI para el uso de NGCGUI
6. Requisitos de archivo para la compatibilidad de NGCGUI
6.1. Requisitos de subrutina de Gcode de archivo único (.ngc)
6.2. Requisitos del archivo Gcode-meta-compiler (.gcmc)
7. Ejemplo DB25

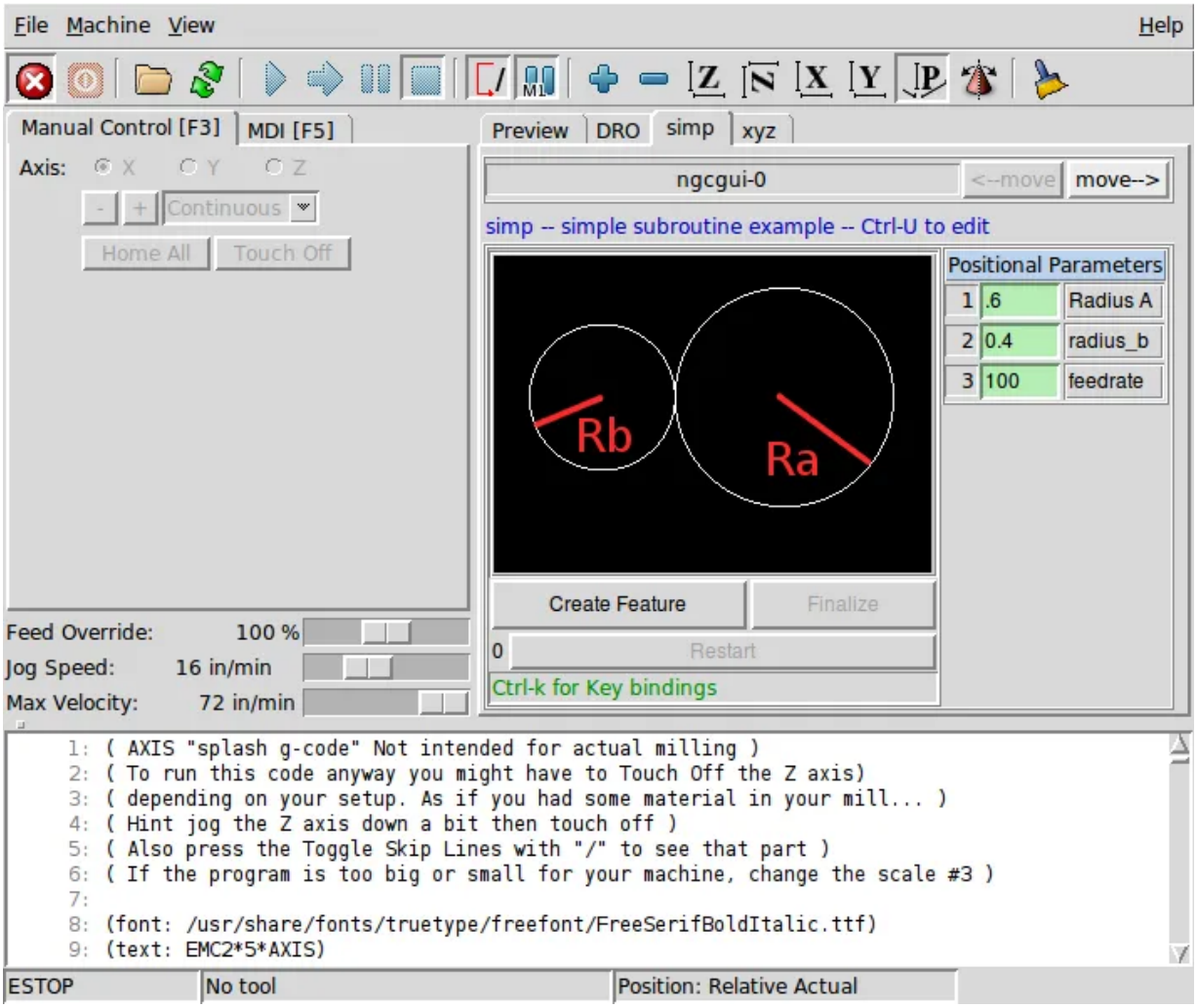


Figura 1. NGCGUI incrustado en Axis

1. Información general

- *NGCGUI* es una aplicación Tcl para trabajar con subrutinas. Le permite tener una interfaz conversacional con LinuxCNC. Puede organizarlas en el orden en que las necesite para ejecutarlas y concatenarlas en un archivo como un programa de pieza completo.
- *NGCGUI* puede ejecutarse como una aplicación independiente o integrarse en varias pestañas en la GUI Axis
- *PYNGCGUI* es una implementación alternativa en Python de ngcgui.
- *PYNGCGUI* puede ejecutarse como una aplicación independiente o puede integrarse como pestañas (con su propio conjunto de pestañas de subrutinas múltiples) en cualquier GUI que admita la incrustación de aplicaciones gladevcp; Axis, Touchy, Gscreen y Gmoccapy.

Usando NGCGUI o PYNGCGUI:

- Se proporcionan pestañas para cada subrutina especificada en el archivo INI
- Se pueden agregar nuevas pestañas de subrutinas sobre la marcha utilizando pestañas personalizada
- Cada pestaña de subrutina proporciona dialogos de entrada para todos los parámetros de la subrutina
- Los dialogos de entrada pueden tener un valor predeterminado y una etiqueta que se identifica mediante comentarios especiales en el archivo de subrutina

- Las invocaciones de subrutinas se pueden concatenar juntas para formar un programa de múltiples pasos
- Se puede usar cualquier subrutina de código G de un solo archivo que se ajuste a las convenciones ngcgui
- Se puede usar cualquier programa gcmc (G code-meta-compiler) que se ajuste a las convenciones ngcgui para etiquetar variables.(El ejecutable de gcmc debe instalarse por separado, consulte: **http://www.vagrearg.org/content/gcmc**)

Nota	NGCGUI y PYNGCGUI implementan las mismas funciones y procesan archivos .ngc y .gcmc que se ajustan a algunas convenciones específicas de ngcgui. En este documento, el término *NGCGUI* generalmente se refiere a cualquier aplicación.

2. Configuraciones de demostración

Varias configuraciones de demostración se encuentran en el directorio sim de las Configuraciones de muestra que ofrece el selector de configuración LinuxCNC. El selector de configuración está en el menú principal del sistema, CNC > LinuxCNC:

Se incluyen ejemplos para Axis, Touchy, Gscreen y Gmoccapy. Estos ejemplos demuestran configuraciones cartesianas de 3 ejes (XYZ) (como fresadoras) y configuraciones de torno (XZ). Algunos ejemplos muestran el uso de un teclado emergente para sistemas de pantalla táctil y otros ejemplos demuestran el uso de archivos creados para la aplicación gcmc (Meta compilador de código G). Los ejemplos Touchy también demuestran la incorporación de un visor de backplot gladevcp (gremlin_view).

La aplicación más simple se encuentra en:

```
Configuraciones_de_muestra/sim/axis/ngcgui/ngcgui_simple
```

Un ejemplo completo que muestra la compatibilidad con gcmc está en:

```
Configuraciones_de_muestra/sim/axis/ngcgui/ngcgui_gcmc
```

Un ejemplo completo integrado como una aplicación gladevcp y usando gcmc está en:

```
Configuraciones_de_muestra/sim/gscreen/ngcgui/pyngcgui_gcmc
```

Las configuraciones sim de ejemplo utilizan archivos de biblioteca que proporcionan ejemplos de archivos de subrutina de código G (.ngc) y archivos de metacompilador de código G (.gcmc):

- *nc_files/ngcgui_lib*
  - *arc1.ngc* : arco básico con compensación del radio de corte
  - *arc2.ngc* : arco especificado por centro, offset, grueso, ángulo (llamado arc1)
  - *backlash.ngc* : rutina para medir el backlash de un eje con un relog comparador
  - *db25.ngc* : crea un corte para conector DB25
  - *gosper.ngc* : una demostración de recursión (flowsnake)
  - *helix.ngc* – hélice o corte con agujero en D
  - *helix_rtheta.ngc* : hélice o agujero en D posicionado por radio y ángulo
  - *hole_circle.ngc* – agujeros en un círculo, igualmente espaciados
  - *ihex.ngc* – hexágono interior
  - *iquad.ngc* – cuadrilátero interior

    - *ohex.ngc* – hexágono exterior
    - *oquad.ngc* – cuadrilátero exterior
    - *qpex_mm.ngc* – demostración de qpockets (en mm)
    - *qpex.ngc* : demostración de qpockets (en pulgadas)
    - *qpocket.ngc* – pocket cuadrado
    - *rectangle_probe.ngc* – sondeo de un área rectangular
    - *simp.ngc* : un ejemplo de subrutina simple que crea dos círculos
    - *slot.ngc* : ranura desde la conexión de dos puntos finales
    - *xyz.ngc* : ejercicio de máquina limitado a una forma de caja
- *nc_files/ngcgui_lib/torno*
    - *g76base.ngc* : interfaz gráfica de usuario para roscado g76
    - *g76diam.ngc* : roscado especificado por diámetros mayor y menor
    - *id.ngc* – *mandrinado* de diámetro interior
    - *od.ngc* : torneado de diámetro exterior
    - *taper-od.ngc* – torneado de cono en diámetro exterior
- *nc_files/gcmc_lib*
    - *drill.gcmc* – perforar agujeros en patrón rectangular
    - *square.gcmc* – demostración simple de etiquetas variables para archivos gcmc
    - *star.gcmc* : demostración de gcmc que ilustra funciones y matrices
    - *wheels.gcmc* : demostración de gcmc de patrones complejos

Para probar una demostración, seleccione una configuración sim e inicie el programa linuxCNC.

Si usa la GUI Axis, presione *E-Stop* , luego *Encender Máquina* y por ultimo *Origen todos* . Elija una pestaña ngcgui, complete los espacios en blanco vacíos con valores sensibles y presione *Crear entidad* y luego *Finalizar* . Finalmente presione la tecla *Ejecutar* para verlo correr. Experimente creando múltiples características y/o desde diferentes pestañas.

Para crear varias subrutinas concatenadas en un solo archivo, vaya a cada pestaña, complete los espacios en blanco, presione *Crear función* y luego, con las teclas de flecha, mueva las pestañas necesarias para ordenarlas. Ahora presione *Finalizar* y responda el mensaje para crearla.

Otras guis tendrán una funcionalidad similar, pero los botones y los nombres pueden ser diferentes.

Nota	Las configuraciones de demostración crean páginas con pestañas solo para algunos de los ejemplos proporcionados. Cualquier interfaz gráfica de usuario con una pestaña personalizada puede abrir cualquiera de las subrutinas de ejemplo de la biblioteca o cualquier archivo de usuario si está en la ruta de la subrutina linuxCNC. Para ver las combinaciones de teclas especiales, haga clic dentro de una página de pestaña ngcgui a enfocar y luego presione Control-k. Las subrutinas de demostración deben ejecutarse en las configuraciones de máquina simulada incluidas en la distribución. El usuario debe comprender el comportamiento y el propósito de un programa antes de ejecutarlo en una máquina real.

3.- Ubicaciones de bibliotecas

En las instalaciones de linuxCNC desde paquetes deb, las configuraciones de simulación para ngcgui usan enlaces simbólicos a bibliotecas LinuxCNC, que no pueden ser escritas por el usuario, para:

- *nc_files/ngcgui_lib* subficheros ngcgui-compatible
- *nc_files/ngcgui_lib/lathe* subficheros de torno ngcgui-compatible
- *nc_files/gcmc_lib* programas ngcgui-compatible
- *nc_files/ngcgui_lib/utilitysubs* Subrutinas de ayuda
- *nc_files/ngcgui_lib/mfiles* Archivos M de usuario

Estas bibliotecas están ubicadas mediante elementos de archivo ini que especifican las rutas de búsqueda utilizadas por linuxCNC (y ngcgui):

```
[RS274NGC]
SUBROUTINE_PATH = ../../nc_files/ngcgui_lib:../../nc_files/gcmc_lib:../../nc_
USER_M_PATH     = ../../nc_files/ngcgui_lib/mfiles
```

Nota	Estas son líneas largas (no en varias líneas) que especifican los directorios utilizados en un parche de búsqueda. Los nombres de directorio están separados por dos puntos (:). No debe haber espacios entre los nombres de directorio.

Un usuario puede crear nuevos directorios para sus propias subrutinas y archivos M y agregarlos a las rutas de búsqueda.

Por ejemplo, un usuario podría crear directorios desde la terminal con los comandos:

```
mkdir /home/mi_nombre_user/mi_subs
mkdir /home/mi_nombre_user/mi_M_files
```

Luego cree o copie los archivos proporcionados por el sistema a estos directorios que el usuario puede escribir. Por ejemplo, un usuario puede crear un subarchivo compatible con ngcgui llamado /home/mi_nombre_user/mi_subs /example.ngc

Para usar archivos en nuevos directorios, el archivo ini debe editarse para incluir los nuevos subarchivos y aumentar las rutas de búsqueda. Para este ejemplo:

```
[RS274NGC]
...
SUBROUTINE_PATH = /home/mi_nombre_user/mi_subs:../../nc_files/ngcgui_lib:.
USER_M_PATH = /home/mi_nombre_user/mi_M_files:../../nc_files/ngcgui_lib/mf

[DISPLAY]
...
NGCGUI_SUBFILE = example.ngc
...
```

LinuxCNC (y ngcgui) usan el primer archivo encontrado al buscar en los directorios de la ruta de búsqueda. Con este comportamiento, puede reemplazar un subarchivo ngcgui_lib colocando un subarchivo con un nombre idéntico en un directorio que se encuentra anteriormente en la búsqueda de ruta. Se puede encontrar más información en el capítulo INI del Manual de integradores.

4. Uso independiente

4.1. NGCGUI independiente

Para su uso, escriba un terminal:

```
ngcgui --help
Uso:
  ngcgui --help |  -?
  ngcgui [Opciones] -D nc_files_directory_name
  ngcgui [Opciones] -i LinuxCNC_inifile_name
  ngcgui [Opciones]

  Opciones:
         [-S subrutina_archivo]
         [-p preámbulo_archivo]
         [-P postamble_archivo]
         [-o archivo_salida]
         [-a autosend_file] (autoenviado a Axis. Predeterminado: auto.ngc)
         [--noauto] (sin autoenviado a Axis)
         [-N |  --nom2] (sin terminacion M2 (usar %))
         [--font [big|small|fontspec]] (predeterminado: "Helvetica -10 norr
         [--horiz | --vert] (predeterminado: --horiz)
         [--cwidth comment_width] (ancho del campo de comentario)
         [--vwidth varname_width] (ancho del campo varname)
         [--quiet] (pocos comentarios en archivo de salida)
         [--noiframe] (predeterminado: el cuadro muestra la imagen)
```

Nota	Como aplicación independiente, ngcgui maneja un único archivo de subrutina que se puede invocar varias veces. Se pueden iniciar múltiples aplicaciones ngcgui independientes de forma independiente.

4.2. PYNGCGUI independiente

Para su uso, escriba un terminal:

```
pyngcgui --help
Uso:
pyngcgui [Opciones] [sub_filename]
Opciones que requieren valores:
    [-d |  --demo] [0|1|2](0: DEMO de nivel superior independiente)
                          (1: DEMO incrustar nuevo cuaderno)
                          (2: DEMO incrustado en el cuaderno existente)
    [-S |  --subfile      sub_filename]
    [-p |  - preamble     preamble_filename]
    [-P |  --postamble    postamble_filename]
    [-i |  --ini          nombre_ini]
    [-a |  --autofile     auto_filename]
    [-t |  --test         testno]
    [-K |  --keyboardfile glade_file] (use un archivo personalizado de tec
Opciones sin valores:
    [-v |  --verbose]
    [-D |  --debug]
    [-N |  --nom2]        (sin terminacion M2 (usar %))
    [-n |  --noauto]      (guardar pero no enviar el resultado automáticame
    [-k |  --keyboard]    (usar teclado emergente predeterminado)
    [-s |  --sendtoaxis]  (envía el archivo ngc generado a la gui Axis)
Notas:
      Un conjunto de archivos se compone de un preámbulo, subarchivo y post
      El preámbulo y el postámbulo son opcionales.
      Se puede especificar un conjunto de archivos desde la linea de comand
      Se pueden especificar múltiples conjuntos de archivos desde un archiv
      Si NO se especifica --ini : buscar un linuxCNC en ejecución y usar su
```

Nota	Como aplicación independiente, pyngcgui puede leer un archivo ini (o una aplicación linuxCNC en ejecución) para crear pestañas para múltiples subarchivos.

5.1. NGCGUI embebido en Axis

Los siguientes elementos del archivo INI van en la sección [DISPLAY]. (Consulte las secciones adicionales a continuación para ver los elementos adicionales necesarios)

- *TKPKG = Ngcgui 1.0* – el paquete NGCGUI
- *TKPKG = Ngcguittt 1.0* : el paquete True Type Tracer para generar texto para grabar (opcional, debe seguir TKPKG = Ngcgui).
- *TTT = truetype-tracer* : nombre del programa truetype tracer (debe estar en la RUTA del usuario)
- *TTT_PREAMBLE = in_std.ngc* – Opcional, especifica el nombre del archivo para el preámbulo utilizado para los archivos sub creados por ttt. (alternativa: mm_std.ngc)

Nota	Los elementos opcionales de truetype tracer se utilizan para especificar una página de pestaña compatible con ngcgui que utiliza la aplicación truetype-tracer. La aplicación truetype-tracer debe instalarse de forma independiente y ubicarse en la RUTA del usuario.

5.2. Incrustar PYNGCGUI como una pestaña gladevcp en una GUI

Los siguientes elementos del archivo INI van en la sección [DISPLAY] para usar con el eje, gscreen o touch guis. (Consulte las secciones adicionales a continuación para ver los elementos adicionales necesarios)

EMBED_ Items

```
EMBED_TAB_NAME = Pyngcgui - nombre que aparecerá en la pestaña incrustada
EMBED_TAB_COMMAND = gladevcp -x {XID} pyngcgui_axis.ui - invoca gladevcp
EMBED_TAB_LOCATION = nombre_de_ubicación - donde se encuentra la página in
```

Nota	El especificador EMBED_TAB_LOCATION no se usa para la GUI del eje. Si bien pyngcgui se puede incrustar en el eje, la integración es más completa cuando se usa ngcgui (usando TKPKG = Ngcgui 1.0). Para especificar EMBED_TAB_LOCATION para otras guis, consulte la sección DISPLAY del Capítulo de configuración de INI.

Nota	El front-end gui truetype tracer no está disponible actualmente para aplicaciones gladevcp.

5.3. Elementos adicionales del archivo INI necesarios para ngcgui o pyngcgui

Los siguientes elementos del archivo INI van en la sección [DISPLAY] para cualquier interfaz gráfica de usuario que incorpore ngcgui o pyngcgui.

- *NGCGUI_FONT = Helvetica -12 normal* : especifica el nombre de fuente, tamaño, normal | negrita
- *NGCGUI_PREAMBLE = in_std.ngc* : archivo de preámbulo que se agregará a las subrutinas. Al concatenar varias invocaciones de subrutinas comunes, este preámbulo solo se agrega una vez. Para máquinas basadas en mm, use mm_std.ngc
- *NGCGUI_SUBFILE = filename1.ngc* : crea una pestaña a partir de la subrutina filename1
- *NGCGUI_SUBFILE = filename2.ngc* : crea una pestaña a partir de la subrutina filename2
- *... etc.*
- *NGCGUI_SUBFILE = gcmcname1.gcmc* – crea una pestaña desde el archivo gcmcname1
- *NGCGUI_SUBFILE = gcmcname2.gcmc* – crea una pestaña desde el archivo gcmcname2
- *... etc.*
- *NGCGUI_SUBFILE = “”* – crea una pestaña personalizada que puede abrir cualquier subrutina en la ruta de búsqueda
- *NGCGUI_OPTIONS = opt1 opt2 ...* – Opciones de NGCGUI
  - *nonew* – no permitir hacer una nueva pestaña personalizada
  - *noremove* : no permite eliminar ninguna pestaña
  - *noauto* : sin autosend (use makeFile, luego guarde o envíe manualmente)
  - *noiframe* : sin imagen interna, muestra imágenes en un widget de nivel superior separado
  - *nom2* – no terminar con M2, usar %. Esta opción elimina todos los efectos secundarios de la terminación M2
- *GCMC_INCLUDE_PATH = dirname1: dirname2* – busca directorios para incluir archivos gcmc

Este es un ejemplo de incrustar NGCGUI en Axis. Las subrutinas deben estar en un directorio especificado por [RS274NGC] SUBROUTINE_PATH. Algunas subrutinas de ejemplo usan otras subrutinas, así que asegúrese de tener las dependencias, si las hay, en un directorio SUBROUTINE_PATH. Algunas subrutinas pueden usar archivos M personalizados que deben estar en un directorio especificado por [RS274NGC] USER_M_PATH.

El Gcode-meta-compiler (gcmc) puede incluir declaraciones como: include (“filename.inc.gcmc”); Por defecto, gcmc incluye el directorio actual que, para linuxCNC, será el directorio que contiene el archivo ini de linuxCNC. Se pueden agregar directorios adicionales al orden de búsqueda de gcmc con el elemento GCMC_INCLUDE_PATH.

Muestra de INI basado en gui Axis

```
[RS274NGC]
...
SUBROUTINE_PATH = ../../nc_files/ngcgui_lib:../../ngcgui_lib/utilitysubs
USER_M_PATH = ../../nc_files/ngcgui_lib/mfiles

[DISPLAY]
TKPKG = Ngcgui 1.0
TKPKG = Ngcguittt 1.0
# Ngcgui debe preceder a Ngcguittt

NGCGUI_FONT = Helvetica -12 normal
# especificar solo nombres de archivo, los archivos deben estar en [RS274N(
NGCGUI_PREAMBLE = in_std.ngc
NGCGUI_SUBFILE = simp.ngc
NGCGUI_SUBFILE = xyz.ngc
NGCGUI_SUBFILE = iquad.ngc
NGCGUI_SUBFILE = db25.ngc
NGCGUI_SUBFILE = ihex.ngc
NGCGUI_SUBFILE = gosper.ngc
# especificar "" para una página de pestaña personalizada
NGCGUI_SUBFILE = ""
#NGCGUI_SUBFILE = "" se usa cuando se especifica el marco de imagen si
#                 se requiere abrir otros archivos
#                 las imágenes se colocarán en una ventana de nivel superi(
NGCGUI_OPTIONS =
#NGCGUI_OPTIONS = opt1 opt2 ...
# items opcionales:
# nonew - no permitir hacer una nueva pestaña personalizada
# noremove: no permite eliminar ninguna pestaña
# noauto - sin envío automático (makeFile, luego enviar manualmente)
# noiframe - sin imagen interna, imagen en el nivel superior separado
GCMC_INCLUDE_PATH = /home/myname/gcmc_includes

TTT = truetype-tracer
TTT_PREAMBLE = in_std.ngc

PROGRAM_PREFIX = ../../nc_files
```

Nota	Lo anterior no es un INI para gui Axis completo: los elementos que se muestran son los utilizados por ngcgui. LinuxCNC requiere muchos elementos adicionales para tener un archivo INI completo.

5.4. Truetype Tracer

Ngcgui_ttt proporciona soporte para truetype-tracer (v4). Crea una pestaña en Axis que permite al usuario crear una nueva pestaña ngcgui después de ingresar texto y seleccionar una fuente y otros parámetros. (Truetype-tracer debe instalarse de forma independiente).

Para incrustar ngcgui_ttt en Axis, especifique los siguientes elementos además de los elementos ngcgui:

```
Elemento: [DISPLAY]TKPKG = Ngcgui_ttt version_number
Ejemplo: [DISPLAY]TKPKG = Ngcgui_ttt 1.0
Nota: Obligatorio, especifica la carga de ngcgui_ttt en una página de pesta
      Debe seguir el elemento TKPKG = Ngcgui.

Elemento: [DISPLAY]TTT = path_to_truetype-tracer
Ejemplo: [DISPLAY]TTT = truetype-tracer
Nota: Opcional, si no se especifica, intenta usar /usr/local/bin/truetype-t
      Especifique con ruta absoluta o como un nombre ejecutable simple
      en cuyo caso el entorno PATH del usuario se usará para encontrar el p

Elemento: [DISPLAY]TTT_PREAMBLE = preamble_filename
Ejemplo: [DISPLAY]TTT_PREAMBLE = in_std.ngc
Nota: Opcional, especifica el nombre del archivo para el preámbulo usado pa
      archivos subttt creados.
```

5.5. Especificaciones de ruta de archivo INI

Ngcgui usa la ruta de búsqueda de linuxCNC para buscar archivos.

La ruta de búsqueda comienza con el directorio estándar especificado por:

```
[DISPLAY]PROGRAM_PREFIX = nombre_directorio
```

seguido de múltiples directorios especificados por:

```
[RS274NGC]SUBROUTINE_PATH = directorio1_nombre:directorio1_nombre:director
```

Los directorios pueden especificarse como rutas absolutas o rutas relativas.

```
Ejemplo: [DISPLAY]PROGRAM_PREFIX = /home/myname/linuxcnc/nc_files
Ejemplo: [DISPLAY]PROGRAM_PREFIX = ~ /linuxcnc/nc_files
Ejemplo: [DISPLAY]PROGRAM_PREFIX = ../../nc_files
```

Una ruta absoluta que comienza con un “/” especifica una ubicación completa del sistema de archivos. Una ruta que comienza con un “~ /” especifica una ruta que comienza desde el directorio de usuario. Una ruta que comienza con “~nombre de usuario /” especifica una ruta que comienza en el directorio del nombre de usuario.

Caminos relativos

Las rutas relativas se basan en el directorio home, que es el directorio que contiene el archivo INI. El uso de rutas relativas puede facilitar la reubicación de las configuraciones, pero requiere una buena comprensión de los especificadores de ruta de Linux.

```
./d0 es lo mismo que d0, por ejemplo, un directorio llamado d0 en el direct
  ../d1 se refiere a un directorio d1 en el directorio padre
   ../../d2 se refiere a un directorio d2 en el padre del directorio padre
   ../../../d3 etc.
```

Se pueden especificar varios directorios con [RS274NGC] SUBROUTINE_PATH separandolos con dos puntos. El siguiente ejemplo ilustra el formato para múltiples directorios y muestra el uso de rutas relativas y absolutas.

Ejemplo de directorios múltiples:

```
[RS274NGC] SUBROUTINE_PATH = ../../nc_files/ngcgui_lib:../../nc_files/ngcgu
```

Esta es una línea larga, no trocee en varias líneas. Cuando linuxCNC y/o ngcgui busca archivos, se utiliza el primer archivo encontrado en la búsqueda.

LinuxCNC (y ngcgui) debe poder encontrar todas las subrutinas, incluidas las rutinas de ayuda que se llaman desde los subfiles ngcgui. Es conveniente colocar los subs de utilidad en un directorio separado como se indica en el ejemplo anterior.

La distribución incluye el directorio ngcgui_lib y los archivos de demostración para preámbulos, subarchivos, postambles y archivos auxiliares. Para modificar el comportamiento de los archivos, puede copiar cualquier archivo y colocarlo en una parte anterior de la ruta de búsqueda. El primer directorio buscado es [DISPLAY] PROGRAM_PREFIX. Puede usar este directorio, pero es una mejor práctica crear uno o más directorios dedicados y colocarlos al principio de [RS274NGC] SUBROUTINE_PATH.

En el siguiente ejemplo, los archivos en /home/myname/linuxcnc/mysubs se encontrarán antes que los archivos en ../../nc_files/ngcgui_lib.

Agregar un directorio de usuarios. Ejemplo:

```
[RS274NGC]SUBROUTINE_PATH = /home/myname/linuxcnc/mysubs:../../nc_files/ngc
```

Los nuevos usuarios pueden intentar inadvertidamente usar archivos que no están estructurados para ser compatibles con los requisitos de ngcgui. Ngcgui probablemente informará numerosos errores si los archivos no están codificados según sus convenciones. Las buenas prácticas sugieren que los subfiles compatibles con ngcgui deben colocarse en un directorio dedicado a ese propósito y que los archivos de preámbulo, postamble y auxiliar deben estar en directorios separados para esquivar los intentos de usarlos como subfiles. Los archivos que no están destinados a ser utilizados como subarchivos pueden incluir un comentario especial: “(not_a_subfile)” para que ngcgui los rechace automáticamente con un mensaje relevante.

5.6. Resumen de los detalles del elemento del archivo INI para el uso de NGCGUI

```
Elemento: [RS274NGC]SUBROUTINE_PATH = dirname1:dirname2:dirname3 ...
Ejemplo:  [RS274NGC]SUBROUTINE_PATH = ../../nc_files/ngcgui_lib:../../nc_f
Nota:     opcional, pero muy útil para organizar subarchivos y archivos de

Elemento: [RS274NGC]USER_M_PATH = dirname1:dirname2:dirname3 ...
Ejemplo:  [RS274NGC]USER_M_PATH = ../../nc_files/ngcgui_lib/mfiles
Nota:     Opcional, necesario para localizar mfiles de usuario personaliza

Elemento: [DISPLAY]EMBED_TAB_NAME = nombre para mostrar en la página de pe
Ejemplo:  [DISPLAY]EMBED_TAB_NAME = Pyngcgui
Nota:     Las entradas: EMBED_TAB_NAME, EMBED_TAB_COMMAND, EMBED_TAB_LOCAT
          definen una aplicación integrada para varias guis linuxCNC

Elemento: [DISPLAY]EMBED_TAB_COMMAND = nombre del programa seguido de argu
Ejemplo:  [DISPLAY]EMBED_TAB_COMMAND = gladevcp -x {XID} pyngcgui_axis.ui
Nota:     Para aplicaciones gladevcp, vea el capitulo GladeVCP

Elemento: [DISPLAY]EMBED_TAB_LOCATION = nombre_de_ubicación
Ejemplo:  [DISPLAY]EMBED_TAB_LOCATION = notebook_main
Nota:     Vea ejemplos de archivos INI para posibles ubicaciones
          No requerido para Axis

Elemento: [DISPLAY]PROGRAM_PREFIX = dirname
Ejemplo:  [DISPLAY]PROGRAM_PREFIX = ../../nc_files
Nota:     Obligatorio y necesario para numerosas funciones de linuxCNC
          Es el primer directorio utilizado en la búsqueda de archivos.

Elemento: [DISPLAY]TKPKG = Ngcgui número_versión
Ejemplo:  [DISPLAY]TKPKG = Ngcgui 1.0
Nota:     Requerido solo para la incrustación de GUI de eje, especifica la
          páginas de pestaña de eje ngcgui

Elemento: [DISPLAY]NGCGUI_FONT = font_descriptor
Ejemplo:  [DISPLAY]NGCGUI_FONT = Helvetica -12 normal
Nota:     Opcional, font_descriptor es un especificador de fuente compatib
          con elementos para fonttype -fontsize fontweight
          El valor predeterminado es: Helvetica -10 normal
          Los tamaños de fuente más pequeños pueden ser útiles para pantal
          Los tamaños de fuente más grandes pueden ser útiles para aplicac
          pantalla táctil

Elemento: [DISPLAY]NGCGUI_SUBFILE = subfile_filename
Ejemplo:  [DISPLAY]NGCGUI_SUBFILE = simp.ngc
Ejemplo:  [DISPLAY]NGCGUI_SUBFILE = square.gcmc
Ejemplo:  [DISPLAY]NGCGUI_SUBFILE = ""
Nota:     Use uno o más elementos para especificar compatible con ngcgui
          subarchivos o programas gcmc que requieren una página de pestaña
          Se creará una pestaña "Personalizada" cuando el nombre de archiv
          Un usuario puede usar una pestaña "Personalizada" para explorar e
          archivos e identificar archivos de preámbulo, subarchivo y postar

Elemento: [DISPLAY]NGCGUI_PREAMBLE = preamble_filename
Ejemplo:  [DISPLAY]NGCGUI_PREAMBLE = in_std.ngc
```

```
Nota:     Opcional, cuando se especifica, el archivo se antepone a un subar
          Los archivos creados con páginas de pestañas "Personalizadas" usa
          especificado con la pagina

Elemento: [DISPLAY]NGCGUI_POSTAMBLE = postamble_filename
Ejemplo:  [DISPLAY]NGCGUI_POSTAMBLE = bye.ngc
Nota:     Opcional, cuando se especifica, el archivo se agrega a un subarch
          Los archivos creados con páginas de pestaña "Personalizadas" usar
          especificado con la pagina

Elemento: [DISPLAY]NGCGUI_OPTIONS = opt1 opt2 ...
Ejemplo:  [DISPLAY]NGCGUI_OPTIONS = nonew noremove
Nota:     las opciones múltiples están separadas por espacios en blanco.
          Por defecto, ngcgui configura páginas de pestañas para que:
            1) un usuario puede hacer nuevas pestañas
            2) un usuario puede eliminar pestañas (excepto la última restar
            3) los archivos finalizados se envían automáticamente a linuxCN
            4) un marco de imagen (iframe) está disponible para mostrar
               una imagen para el subarchivo (si se proporciona una imagen)
            5) el archivo de resultados ngcgui enviado a linuxCNC finaliza
               un m2 (e incurre en efectos secundarios de m2)
          Las opciones nonew, noremove, noauto, noiframe, nom2 respectivame
          deshabilitar estos comportamientos predeterminados.
          Por defecto, si un archivo de imagen (.png, .gif, jpg, pgm)
          se encuentra en el mismo directorio que el subarchivo, la
          imagen se muestra en el iframe.  Especificando la
          opción noiframe pone a disposición botones adicionales
          para seleccionar un preámbulo, subarchivo y postámbulo y
          casillas de verificación adicionales. La Seleccion de casillas de
          verificación siempre están disponibles con teclas especiales:
            Ctrl-R Toggle "Retener valores en la lectura del subarchivo"
            Ctrl-E Alternar "Expandir subrutina"
            Ctrl-a Alternar "Envío automático"
            (Ctrl-k enumera todas las teclas y funciones)
          Si se especifica noiframe y se encuentra un archivo de imagen,
          la imagen se muestra en una ventana separada y
          todas las funciones están disponibles en la página de pestañas.
          Las NGCGUI_OPTIONS se aplican a todas las pestañas ngcgui, pero l
          opciones nonew, noremove y noiframe no son aplicables a pestañas
          "Personalizadas".  No use pestañas "Personalizadas" si quiere
          limitar la capacidad del usuario para seleccionar subarchivos o c
          pestañas adicionales.

Elemento: [DISPLAY]GCMC_INCLUDE_PATH = dirname1:dirname2:...
Ejemplo:  [DISPLAY]GCMC_INCLUDE_PATH = /home/myname/gcmc_includes:/home/myn
Nota:     Opcional, cada directorio se incluirá cuando se invoque gcmc
          usando la opción: --incluir dirname
```

6. Requisitos de archivo para la compatibilidad de NGCGUI

6.1. Requisitos de subrutina de Gcode de archivo único (.ngc)

Un subfile compatible con NGCGUI contiene una única definición de subrutina. El nombre de la subrutina debe ser el mismo que el nombre del archivo (sin incluir el sufijo .ngc). LinuxCNC admite subrutinas con nombre o numeradas, pero solo las subrutinas con nombre son compatibles con NGCGUI. Para obtener más información, consulte el capítulo O-Code.

La primera línea sin comentarios debe ser una declaración *sub*. La última línea sin comentarios debe ser una declaración *endsub*.

examp.ngc:

```
(información: texto_info_encima_de_pestaña_de_pagina)
;  línea de comentario que comienza con punto y coma
(línea de comentario usando paréntesis)
o<examp> sub
  CUERPO_DE_SUBRUTINA
o<examp> endsub
;  línea de comentario que comienza con punto y coma
(línea de comentario usando paréntesis)
```

El cuerpo de la subrutina debe comenzar con un conjunto de sentencias que definen los parámetros con nombre locales para cada parámetro posicional esperado para la llamada a la subrutina. Estas definiciones deben ser consecutivas comenzando con #1 y terminando con el último número de parámetro utilizado. Se deben proporcionar definiciones para cada uno de estos parámetros (sin omisiones).

Numeración de Parámetros

```
#<xparam>  = #1
#<yparam>  = #2
#<zparam>  = #3
```

LinuxCNC considera que todos los parámetros numerados en el rango #1 al # 30 son parámetros de llamada, por lo que ngcgui proporciona cuadros de entrada para cualquier ocurrencia de parámetros en este rango. Es una buena práctica evitar el uso de los parámetros numerados del 1 al 30 en cualquier otro lugar de la subrutina. Se recomienda el uso de parámetros locales con nombre para todas las variables internas.

Cada declaración de definición puede incluir opcionalmente un comentario especial y un valor predeterminado para el parámetro.

Prototipo de declaración

```
#<vname>  = #n (= default_value)
o
#<vname>  = #n (comentario_texto)
o
#<vname>  = #n (= default_value comment_text)
```

Ejemplos de parámetros

```
#<xparm> = #1 (=0.0)
#<yparm> = #2 (Ystart)
#<zparm> = #3 (=0.0 Z start setting)
```

Si se proporciona un valor predeterminado, se ingresará en el cuadro de entrada para el parámetro al inicio.

Si se incluye comentario_texto, se usará para identificar la entrada en lugar del nombre del parámetro.

Parámetros nombrados globales

Notas sobre parámetros globales con nombre y ngcgui:

(los parámetros globales con nombre tienen un guión bajo en el nombre, como #<_someglobalname>)

Como en muchos lenguajes de programación, el uso de globales es poderoso, pero a menudo puede tener consecuencias inesperadas. En LinuxCNC, los parámetros con nombre globales existentes serán válidos en la ejecución de la subrutina y las subrutinas pueden modificar o crear parámetros con nombre global.

Se desaconseja pasar información a las subrutinas que utilizan parámetros con nombre global, ya que dicho uso requiere el establecimiento y mantenimiento de un contexto global bien definido que es difícil de mantener. El uso de los parámetros numerados del 1 al 30 como entradas de subrutina debería ser suficiente para satisfacer una amplia gama de requisitos de diseño.

Si bien no se recomiendan los parámetros con nombre globales de entrada, las subrutinas linuxCNC deben usar parámetros con nombre globales para devolver resultados. Dado que los subfiles compatibles con ngcgui están destinados al uso de gui, los valores de retorno no son un requisito común. Sin embargo, ngcgui es útil como herramienta de prueba para subrutinas que devuelven parámetros con nombre globales y es común que los subfiles compatibles con ngcgui llamen a archivos de subrutinas de utilidad que devuelven resultados con parámetros con nombre globales.

Para admitir estos usos, ngcgui ignora los parámetros globales con nombre que incluyen un carácter de dos puntos (:) en su nombre. El uso de los dos puntos (:) en el nombre evita que ngcgui cree cuadros de entrada para estos parámetros.

Parámetros nombrados globales

```
o<examp> sub
...
#<_examp:result> = #5410       (devuelve el diámetro actual de la herramienta
...
o<helper> call [#<x1>] [#<x2>] (llamar a una subrutina)
#<xresult> = #<_helper:answer> (localizar inmediatamente el resultado global
#<_helper:answer> = 0.0        (anula el parámetro global con nombre utilizad
...
o<examp> endsub
```

En el ejemplo anterior, la subrutina de la utilidad se encontrará en un archivo separado llamado helper.ngc. La rutina auxiliar devuelve un resultado en un parámetro con nombre global llamado #<_ helper: answer.

Para una buena práctica, el subfile que realiza la llamada localiza inmediatamente el resultado para usarlo en otra parte del subfile y el parámetro con nombre global utilizado para devolver el resultado se anula en un intento de mitigar su uso involuntario en otro lugar del contexto global. (Un valor de anulación de 0.0 puede no ser siempre una buena opción).

Ngcgui admite la creación y concatenación de múltiples características para un subfile y para múltiples subfiles. A

veces es útil para los subarchivos determinar su orden en tiempo de ejecución, por lo que ngcgui inserta un parámetro global especial que se puede probar dentro de las subrutinas. El parámetro se llama # . Su valor comienza con un valor de 0 y se incrementa para cada característica agregada.

Características adicionales

Se puede incluir un comentario de *información* especial en cualquier parte de un subfile compatible con ngcgui. El formato es:

```
(info: info_text)
```

Info_text se muestra cerca de la parte superior de la página de pestaña ngcgui en el eje.

Los archivos que no están destinados a usarse como subarchivos pueden incluir un comentario especial para que ngcgui los rechace automáticamente con un mensaje relevante.

```
(not_a_subfile)
```

Un archivo de imagen opcional (.png, .gif, .jpg, .pgm) puede acompañar a un subarchivo. El archivo de imagen puede ayudar a aclarar los parámetros utilizados por el subarchivo. El archivo de imagen debe estar en el mismo directorio que el subfile y tener el mismo nombre con un sufijo de imagen apropiado, por ejemplo, el subfile example.ngc podría estar acompañado por un archivo de imagen examp.png. Ngcgui intenta cambiar el tamaño de las imágenes grandes submuestreando un tamaño con un ancho máximo de 320 y una altura máxima de 240 píxeles.

Ninguna de las convenciones requeridas para hacer un subfile compatible con ngcgui impide su uso como archivo de subrutina de propósito general para LinuxCNC.

La distribución LinuxCNC incluye una biblioteca (directorio ngcgui_lib) que incluye subfiles y archivos de utilidad compatibles con ngcgui de ejemplo para ilustrar las características de las subrutinas LinuxCNC y el uso de ngcgui. Otra biblioteca (gcmc_lib) proporciona ejemplos de archivos de subrutina para el metacompilador Gcode (gcmc)

Se pueden encontrar subrutinas adicionales para usuarios en el Foro en la sección Subrutinas.

6.2. Requisitos del archivo Gcode-meta-compiler (.gcmc)

Ngcgui lee los archivos para el metacompilador Gcode (gcmc) y crea cuadros de entrada para las variables etiquetadas en el archivo. Cuando se finaliza una función para el archivo, ngcgui pasa el archivo como entrada al compilador gcmc y, si la compilación es exitosa, el archivo gcode resultante se envía a linuxCNC para su ejecución. El archivo resultante está formateado como subrutina de un solo archivo; ngcgui puede mezclar archivos .gcmc y archivos .ngc.

Las variables identificadas para su inclusión en ngcgui están etiquetadas con líneas que aparecerán como comentarios para el compilador gcmc.

Ejemplos de formatos de etiquetas variables

```
// ngcgui: varname1 =
// ngcgui: varname2 = value2
// ngcgui: varname3 = value3, label3;
```

Ejemplos:

```
// ngcgui: zsafe =
// ngcgui: feedrate = 10
// ngcgui: xl = 0, límite x
```

Para estos ejemplos, el cuadro de entrada para varname1 no tendrá un valor predeterminado, el cuadro de entrada para varname2 tendrá un valor predeterminado de valor2 y el cuadro de entrada para varname 3 tendrá un valor predeterminado de 3 y una etiqueta label3 (en lugar de varname3). Los valores predeterminados deben ser números.

Para facilitar la modificación de líneas válidas en un archivo gcmc, se aceptan formatos alternativos de línea de etiqueta. Los formatos alternativos ignoran los puntos y comas finales (;) y los marcadores de comentarios finales (//) Con esta disposición, a menudo permite agregar *//ngcgui:etiqueta* a las líneas existentes en un archivo .gcmc.

Formatos de etiquetas variables alternativas

```
// ngcgui: varname2 = value2;
// ngcgui: varname3 = value3;  //, etiqueta3;
```

Ejemplos:

```
// ngcgui: feedrate = 10;
// ngcgui: xl = 0;  //, límite x
```

Una línea de información que aparecerá en la parte superior de una página de pestaña puede incluirse opcionalmente con una línea etiquetada como:

Etiqueta de información

```
//ngcgui: info: text_to_appear_at_top_of_tab_page
```

Cuando sea necesario, las opciones se pueden pasar al compilador gcmc con una línea etiquetada:

Formato de etiqueta de línea opcional

```
//ngcgui: -option_name [[=] option_value]
```

Ejemplos:

```
// ngcgui: -I
// ngcgui: --imperial
// ngcgui: --precision 5
// ngcgui: --precision=6
```

Las opciones para gcmc están disponibles con el comando de terminal:

```
gcmc --help
```

Un programa gcmc por defecto usa el modo métrico. El modo se puede configurar en pulgadas con la

configuración de la opción:

```
//ngcgui: --imperial
```

Un archivo de preámbulo, si se utiliza, puede establecer un modo (g20 o g21) que entre en conflicto con el modo utilizado por un archivo gcmc. Para asegurarse de que el modo de programa gcmc esté vigente, incluya la siguiente declaración en el archivo .gcmc:

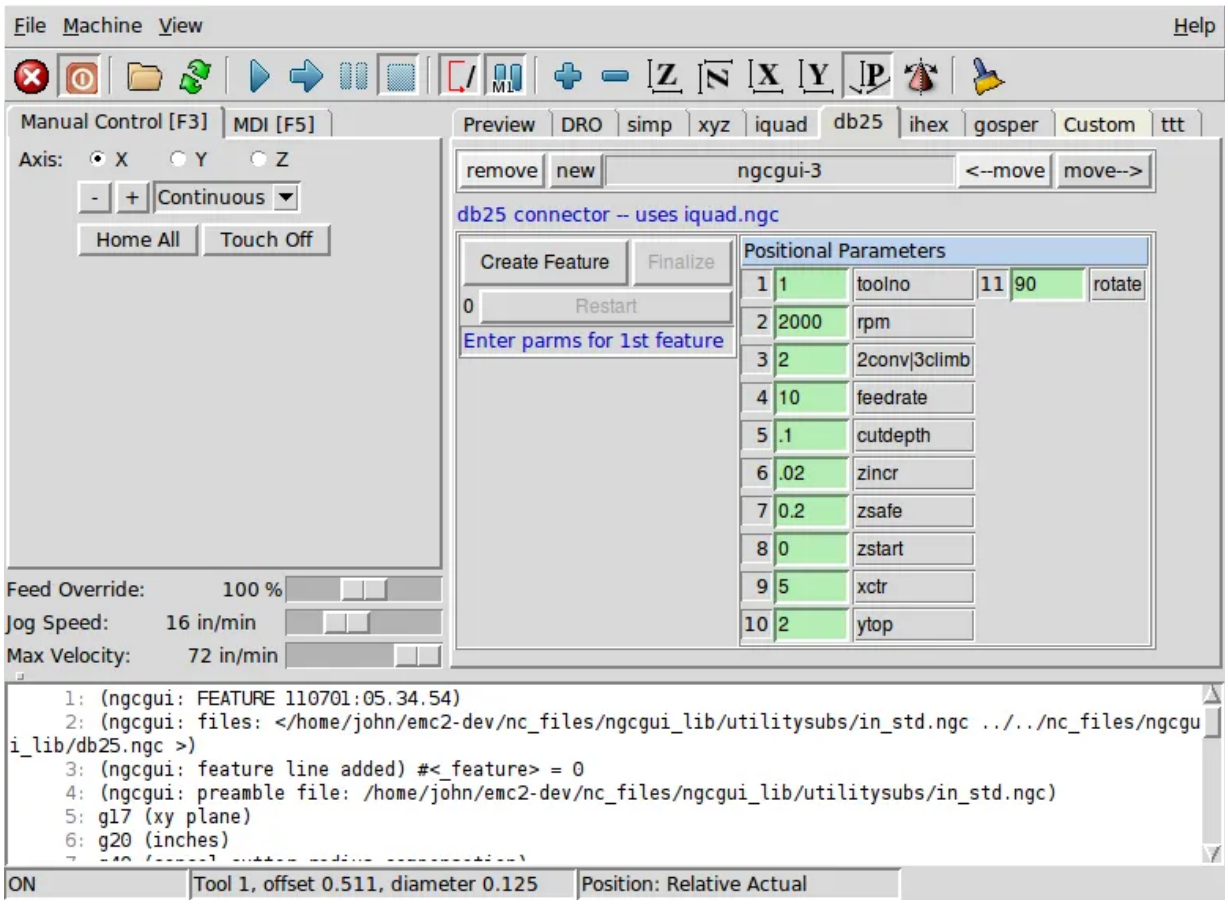
```
include("sure_mode.gcmc")
```

y proporcione una ruta adecuada para gcmc include_files en el archivo ini, por ejemplo:

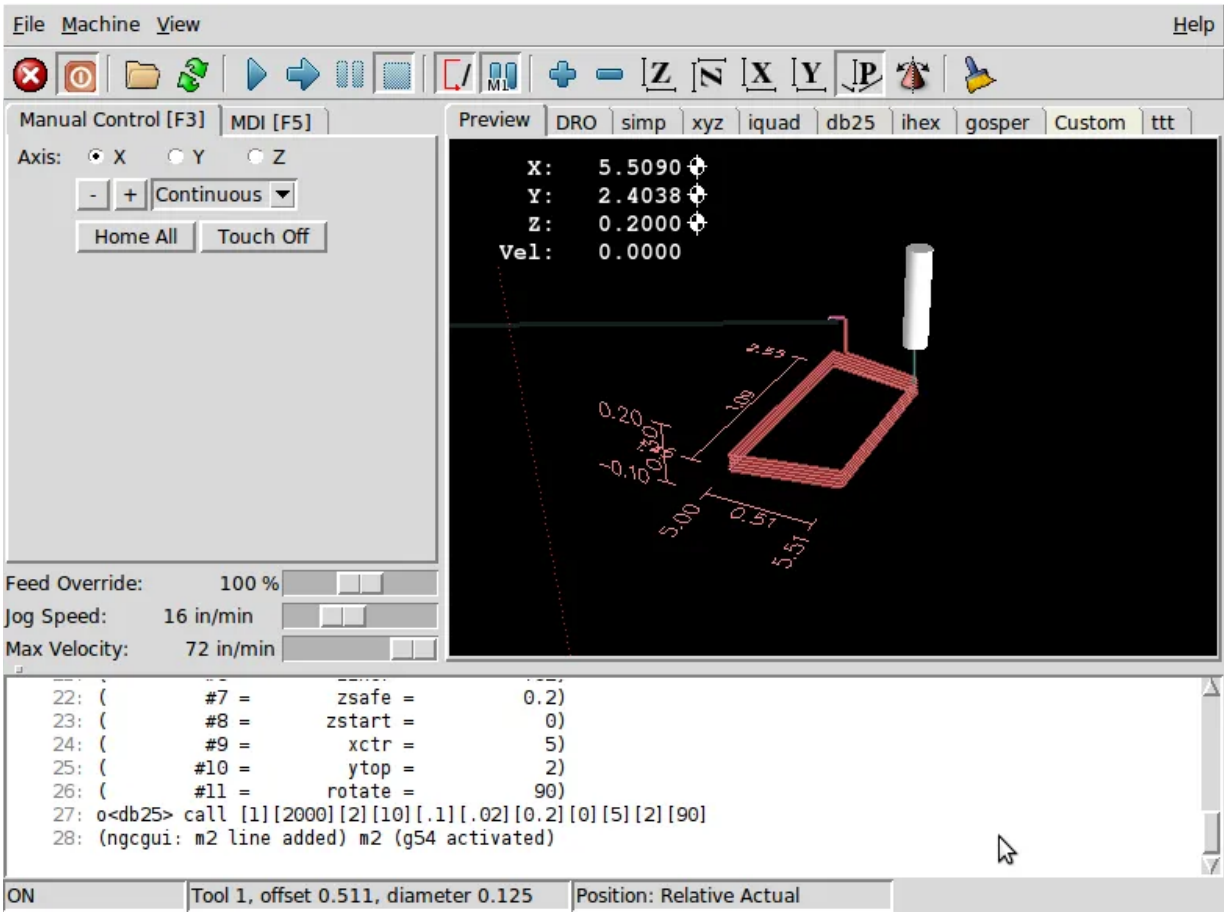
```
[DISPLAY]
GCMC_INCLUDE_PATH = ../../nc_files/gcmc_lib
```

7. Ejemplo DB25

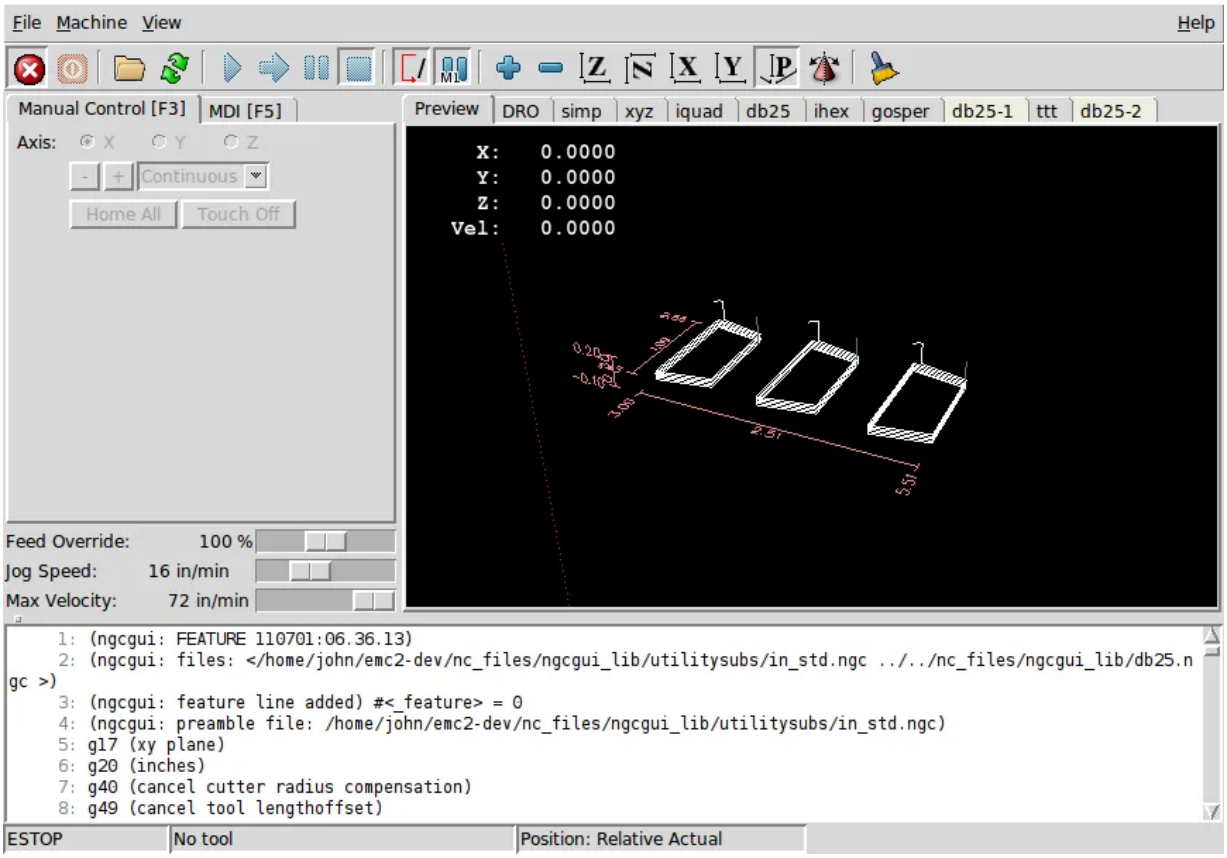
A continuación se muestra la subrutina DB25. En esta primera captura, verá dónde se rellenan los espacios en blanco para cada variable.



Esta captura muestra el backplot de la subrutina DB25.



Esta captura muestra el uso del nuevo botón y la pestaña personalizada para crear tres recortes DB25 en un programa.



Última actualización 2018-10-31 16:23:23 MDT

Relacionado

Lanzador de configuraciones
En "Sin categoría"

Linux FAQ
En "Doc. oficial"

Construyendo LinuxCNC; Autotools
En "Linuxcnc avanzado"

linuxcnces **12 noviembre, 2019** **Doc. oficial** **Editar**

LinuxCÑC,