

Prototipo de Impresora 3D reutilizando desecho electrónico (e-waste) en su construcción, y uso de botellas PET como material de impresión

Paulo Roberto Loma Marconi
prlomarconi@gmail.com

ABSTRACT

Implementación de un prototipo básico de Impresora 3D con la técnica FDM aplicando botellas plásticas PET desechables reemplazando el tradicional PLA/ABS como material de impresión, adaptación de desecho electrónico (e-waste) para la construcción del extrusor y la estructura. Implementación de diseño propio de electrónica y Firmware-Software para el control de temperatura PID por Efecto Joule, control de motores paso a paso (mPaP), operación y monitoreo general, algoritmos descriptor Gcode y posicionamiento 3D.

Keywords – Impresora 3D, FDM, Manufactura por Adición, Matlab, Visual Studio, C/C#, extrusor, DIY, Efecto Joule, Termocupla, Control PID discreto, Sketchup CAD, e-waste, adaptación, PET, reciclado, descriptor Gcode, RTOS, PIC18F4550, PICCS, USB modo bulk.

I. INTRODUCCIÓN

Durante el proceso de construcción de una impresora 3D existen dos problemas que fueron identificados; la importación de componentes electrónicos (controladores) y electromecánicos (extrusor) específicos para impresoras 3D, y la dependencia en la importación del material de impresión plástico tipo PLA/ABS.

Por tanto, el objetivo fue la implementación de un prototipo básico alternativo de Impresora 3D reutilizando y adaptando desecho electrónico (*e-waste*) en el proceso de construcción, además del empleo de botellas PET desechables como material de impresión, aplicando la técnica FDM (Modelación por Deposición Fundida).

II. DESARROLLO DEL PROTOTIPO

La figura 1 describe en diagrama de bloques el esquema general de funcionamiento para su respectiva implementación.

A. Construcción mecánica del extrusor y estructura completa.

Se dispuso la adaptación de *e-waste* y componentes de fácil adquisición en tiendas locales, para ninguna etapa se recurrió a la importación de componentes exclusivos para Impresoras 3D.

La construcción empezó con la elección de los componentes y herramientas electromecánicas, luego se realizó el modelo CAD del extrusor utilizando la herramienta Sketchup, figuras 2 y 3.

B. Alimentación del material

El tipo de alimentación del material plástico PET picado es manual, luego es transportado por inyección hacia el Licuefactor, figura 4.

También puede usarse rollo de plástico PET obtenido directamente de la botella gracias a un procedimiento manual de corte, figura 5.

C. Calentamiento por Efecto Joule

Se realiza la construcción del Licuefactor con un *nozzle* (boquilla) de bronce para hornillas a gas y una resistencia eléctrica tipo niquelina, todo debe ser aislado de forma eléctrica y calorífica, figura 6.

D. Extrusión

Se aplica la técnica por Inyección donde la barra rosca esta acoplada al extrusor, figura 2; que transfiere el movimiento rotacional del motor mPaP en movimiento vertical lineal.

E. Robot cartesiano y posicionamiento 3D

La estructura completa del prototipo respeta la configuración del robot cartesiano, para cada eje existe un mPaP unipolar o bipolar. En los 3 casos, el eje y/o caja de engranajes del motor tiene acoplado una polea dentada que transfiere el movimiento rotacional en movimiento lineal, las barras lisas permiten que el movimiento lineal

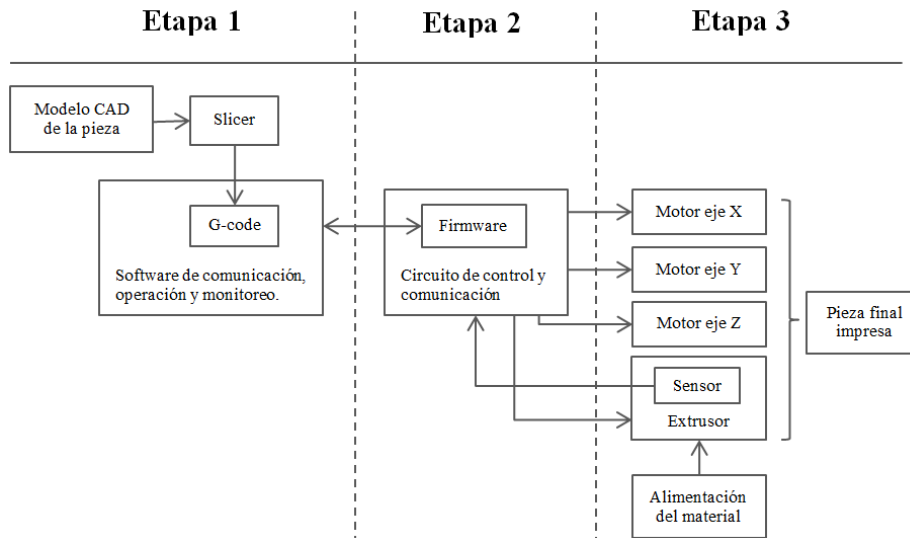


Figura 1: Esquema general de funcionamiento del prototipo.

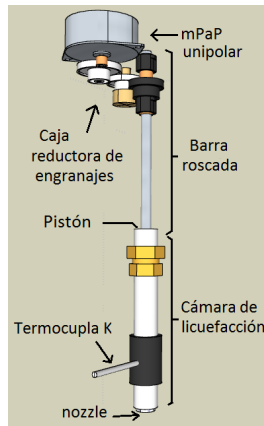


Figura 2: Modelo CAD del extrusor.

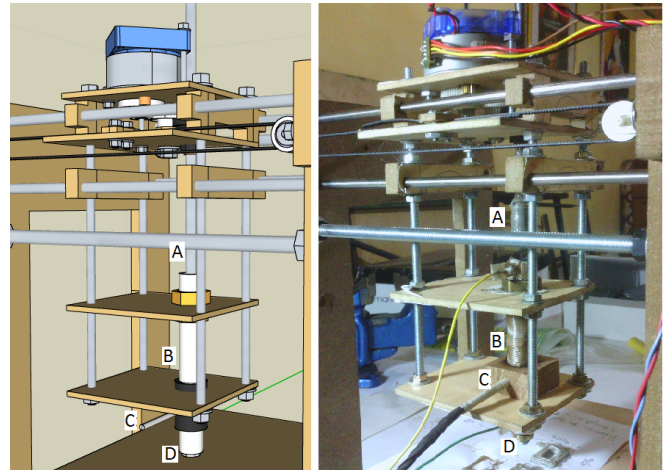
uniforme reduzca su fricción, y las abrazaderas conectan eje con eje para completar la configuración de robot cartesiano, figura 8.

F. Diseño electrónico

El diseño electrónico completo (*Schematic*), *layout* de la placa PCB y modelo CAD, se realizó en Proteus 8.0.

1) MCU PIC18F4550 y comunicación USB

La ejecución del *Firmware* está a cargo del MCU PIC18F4550, cuya característica principal es el módulo USB 2.0 integrado, además de su fácil adquisición y bajo costo. El tipo de transferencia USB que se aplica es *Bulk mode* con tipo de clase CC(*Custom Class*), figura 9.



A: Pistón B: Licuefactor C: Termocupla K D: Nozzle

Figura 3: Extrusor construido y ensamblado.

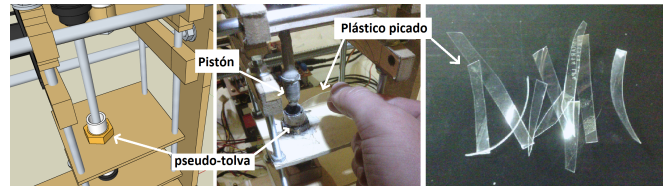


Figura 4: Alimentación del material plástico PET.

2) Esquema electrónico de los cuatro motores mPaP X, Y, Z y E

Se usan cuatro mPaP *e-waste* de diferentes características y dimensiones, un bipolar para el eje X; dos unipolares para el eje Y, Z y un unipolar para el extrusor E. El controlador es de medio paso para todos los motores,



Figura 5: Rollo de plástico PET.



Figura 6: Licuefactor eléctrica-caloríficamente aislado.

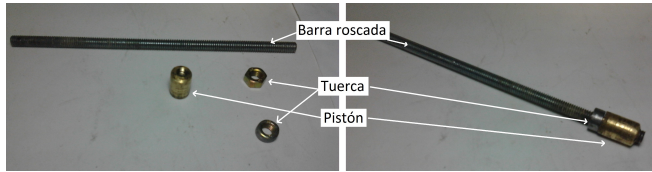


Figura 7: Construcción del pistón.

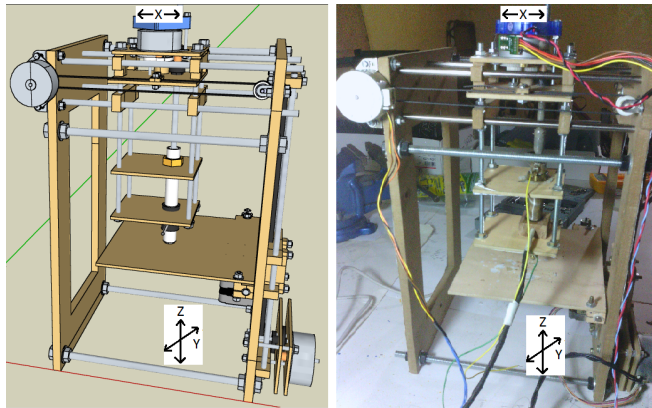


Figura 8: Disposición mecánica de posicionamiento 3D.

y está a cargo de IC L297+IC L298, para el motor bipolar; y IC L297+array *Darlington*, para los motores unipolares; figura 10.

3) Efecto Joule por PWM y Mosfet

Para el calentamiento por Efecto Joule en el Licuefactor, se aplica control PID por PWM desde el MCU 18F4550, cuyo actuador es un Mosfet de potencia IRL3803, conectado en paralelo con una fuente de vol-

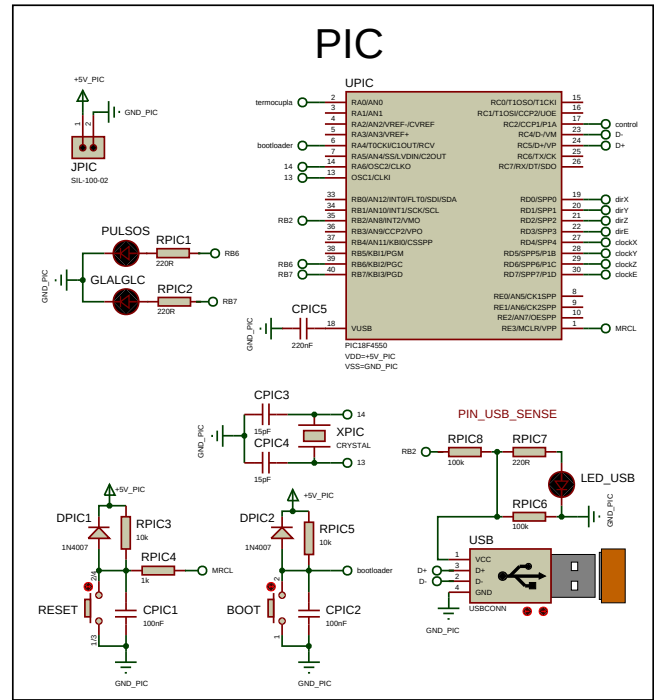


Figura 9: Esquemático electrónico MCU PIC18F4550.

taje de +19,5[V] y 4[A] actuando sobre la niquelina de 6[Ω], otorgando una potencia de 63[watts], figura 11.

4) Acondicionamiento de señal de Termocupla K

Para lograr el acondicionamiento y su compensación analógica correctamente, se debe plantear el problema según la tabla 1, seguido del análisis y obtención de valores mediante la figura 12 y desarrollo matemático (1)

	Temperatura	Voltaje	Sensibilidad
Termocupla K	0 – 300[°C]	0 – 5[V]	$S_k = 40,5 \left[\frac{uV}{^{\circ}C} \right]$
LM35	0 – 150[°C]	0 – 5[V]	$\alpha = 10,0 \left[\frac{mV}{^{\circ}C} \right]$

Tabla 1: Condiciones de implementación.

$$S = \frac{V_{max} - V_{min}}{T_{max} - T_{min}}$$

$$S_1 = 33,3 \left[\frac{mV}{^{\circ}C} \right]; \quad S_2 = 16,7 \left[\frac{mV}{^{\circ}C} \right] \quad (1)$$

bloque de ganancias,

$$G_1 = \frac{S_1}{\alpha} = 3,33; \quad G_2 = \frac{S_1}{\alpha} = 412 \quad (2)$$

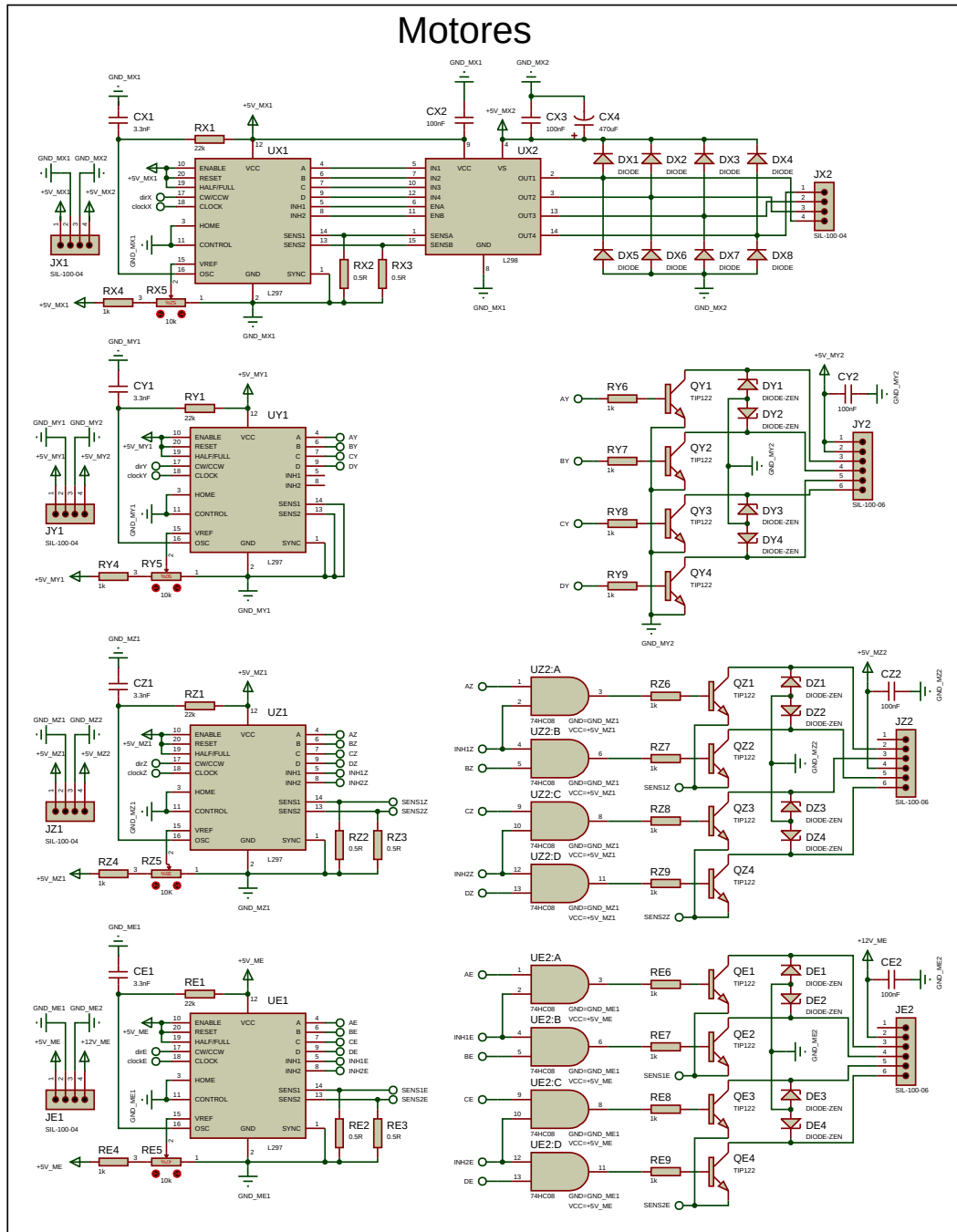


Figura 10: Esquemático completo de control mPaP.

obteniendo valores de Resistencia para el **Amplificador Diferencial(OP07) = UT2** de Ganancia G_1

$$v_o = \frac{R_2}{R_1} (v_2 - 0); \quad G_1 = \frac{v_o}{v_2} = \frac{R_2}{R_1} = 3,33$$

$$R_2 = 330[\Omega]; \quad R_1 = 100[\Omega] \quad (3)$$

reemplazando en el diseño de la figura 13

$$RT2 = RT4 = 330[\Omega]; \quad RT1 = RT3 = 100[\Omega] \quad (4)$$

calculando los valores de Resistencia para el **Amplificador Diferencial de Instrumentación(AD620) = UT3** de Ganancia G_2 , según la ecuación siguiente, y reemplazando en el diseño de la figura 13

$$R_G = \frac{49,4[k\Omega]}{G_2 - 1} = 120[\Omega]; \quad R_{GT} = 120[\Omega]$$

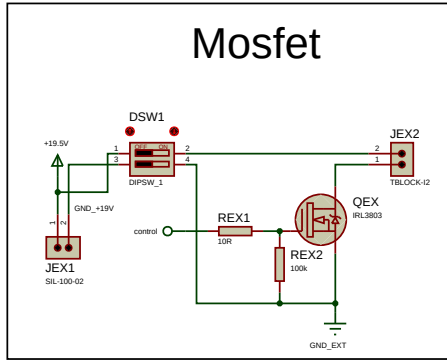


Figura 11: Actuador, Mosfet de potencia.

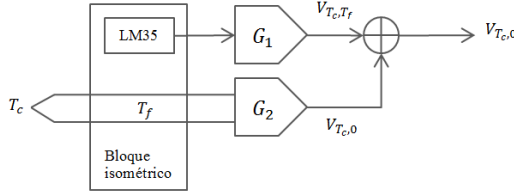


Figura 12: Acondicionamiento de Termocupla.

Para completar el último paso, se debe aplicar la **Ley de Circuitos Homogéneos y Metales Intermedios** según,

$$V_{T_c,0} = V_{T_c,T_f} + V_{T_f,0} \quad (5)$$

y calculando los valores de Resistencia para el **sumador UT4**,

$$v_o = \left(1 + \frac{R_2}{R_1}\right) \frac{\frac{v_1}{R_1} + \frac{v_2}{R_2}}{\frac{1}{R_1} + \frac{1}{R_2}}$$

si, $R_1 = R_2 = 10[k\Omega]$ entonces,

$$v_o = v_1 + v_2; \quad v_{UT4} = v_{UT2} + v_{UT3} \quad (6)$$

5) Modelo CAD y grabado PCB

Finalmente, se construye el PCB respetando el área máxima de soporte; una fuente de poder para PC, y reutilizando la mayor cantidad de componentes electrónicos *e-waste* de placas dañadas, figura 14.

III. DESARROLLO DE FIRMWARE

Programación Imperativa y Estructurada, la razón se debe a la aplicación del Lenguaje de Programación C sobre *software* compilador PICWHD con su respectivo *bootloader*, además de la técnica Multitarea a través de un RTOS *open-source*; (OSA-RTOS) para PIC18F4550.

A. Estructura general

El algoritmo 1 describe en pseudo-código la estructura base. En **config** hay 4 grupos grandes delimitados, MCU y Comunicación USB que están implícitamente relacionados; en cambio, RTOS depende del correcto funcionamiento de su propio grupo de instrucciones y la correspondiente compatibilidad con el MCU.

Hay que resaltar la omisión del clásico **while loop** en **main**; se prescinde de él porque un RTOS gestiona mejor las tareas de operación, control y comunicación.

Algoritmo 1: Estructura general.

Input: Puertos I/O

Output: Actuadores

config

```
MCU;
RTOS;
Comunicación USB;
Control;
```

def

```
variables;
funciones;
tareas RTOS;
```

main

```
call funciones;
call tareas RTOS;
```

B. Función genérica: Inicio

Es la primera función genérica a llamar que inicializa las variables y módulos del MCU, algoritmo 2.

C. Tarea RTOS: Comunicación USB/descriptor Bulk-mode

La figura 15 muestra en diagrama de bloques los Bytes de Envío/Recepción de paquetes USB que implementa a través del algoritmo 3.

D. Tarea RTOS: Control PID discreto de temperatura

Para implementar Control PID se debe obtener el modelo de la Planta (Licuefactor) $G_p(s)$, se analiza los requerimientos de diseño, simulación en **Matlab** y finalmente, según el algoritmo 4 se implementa el controlador PID.

1) Análisis

El *Software* GUI cumple la función de obtener la respuesta al escalón del Licuefactor $G_p(s)$, registrando en un **documento.txt** los valores de sensor de Temperatura Termocupla K cada Periodo de Tiempo de muestreo

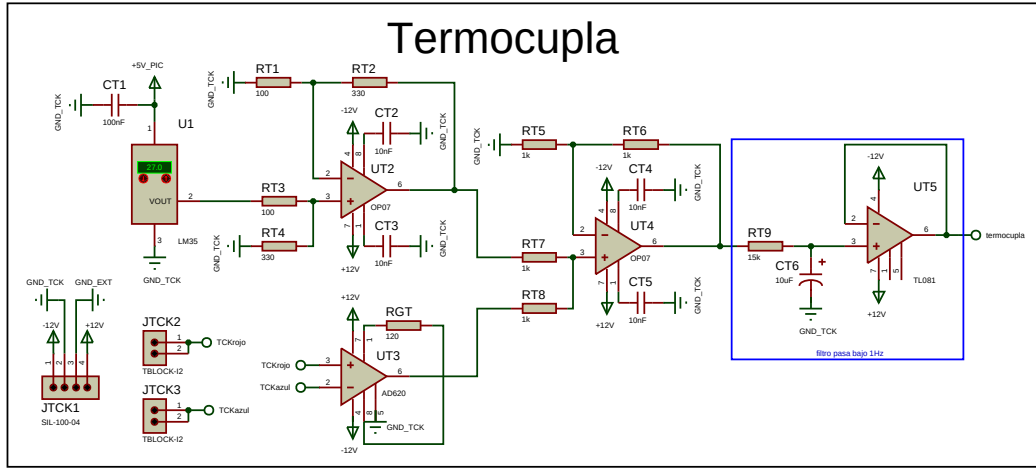


Figura 13: Sensor de Temperatura Termocupla K.

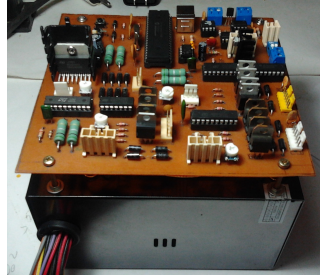
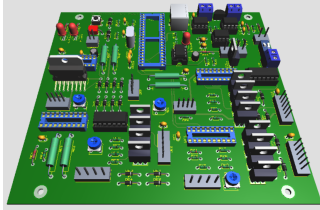


Figura 14: Implementación electrónica completa.

Algoritmo 2: Función genérica: Inicio.**Function** Inicio()**ini** valores mPaP

número de Pulsos XYZE = 0;
 periodo de Pulsos XYZE = 0;

ini constantes PID

$i1 = e1 = d1 = 0$;
 $K_d = 4,372$ $K_p = 4,387$ $K_i = 0,01457$;
 Tiempo de muestreo: $T = 10[s]$;
 $K_{pz} = K_p$ $K_{iz} = K_i \frac{T}{2}$ $K_{dz} = \frac{K_d}{T}$;

config CCP

Frecuencia_PWM = 3kHz;

config ADC

Canal 0 = AN0;

config Timer0 para OsaRTOS

Frecuencia = 1kHz para clock 48MHz;

config USB

Habilitar módulo USB;
 Enumeración USB;

Algoritmo 3: Tarea RTOS: Comunicación USB.**Function** tarea USB()**if** Recibir Byte > 0 **then**

número de Pulsos XYZE = Recibir Byte[0:7];
 periodo de Pulsos XYZE = Recibir Byte[8:13];
 DirMotor XYZE = Recibir Byte[14:17];
 dato $G_{la} - G_{lc}$ = Recibir Byte[18];
 referencia $r(kT)$ PWM = Recibir Byte[19];

if EnviarByte > 0 **then**

dato ADC[0-255] = Recibir Byte[0];
 dato ADC[256-1023] = Recibir Byte[1];
 estado mPaP XYZE = Recibir Byte[2];

delay = 100 [ms];

$T_m[ms]$, la figura 15 muestra el procesamiento de los datos $r(t)$, $y(t)$ y actuador por PWM.

Aplicando una señal escalón a G_p a lazo abierto G_{la} se obtiene la figura 16.

Que exporta un documento.txt con $T_m = 1000[ms]$ para ser importado con la herramienta Ident Matlab, y proceder a la identificación del modelo SISO.

Finalmente genera la Función de Transferencia,

$$G_p(s) = \frac{1,4958}{1 + 429,15 s} \quad (7)$$

2) Diseño

Gracias a otra herramienta de diseño de controlador; Sisotool Matlab, se genera los parámetros y Función de Transferencia del Controlador $G_c(s)$, figura 18

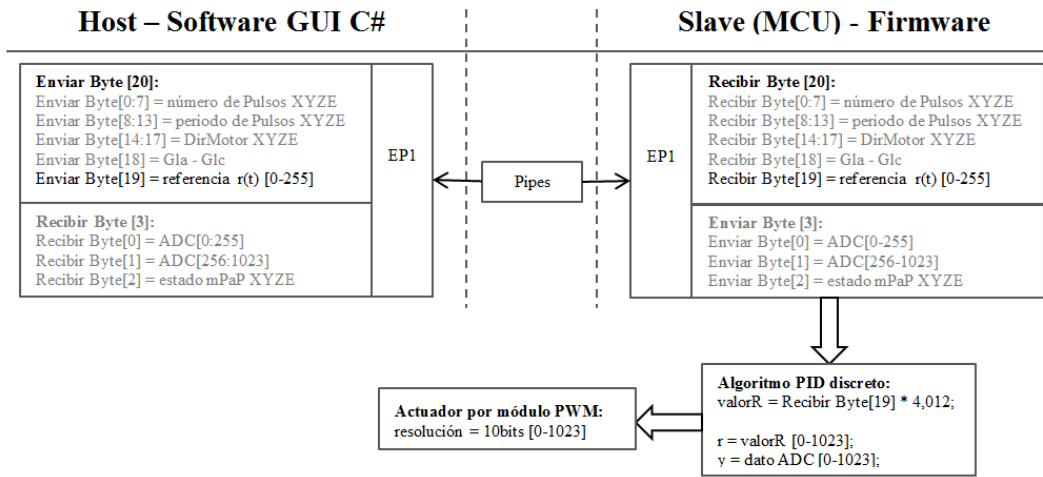


Figura 15: Envío/Recepción de paquetes USB, y procesamiento del PWM para control PID.

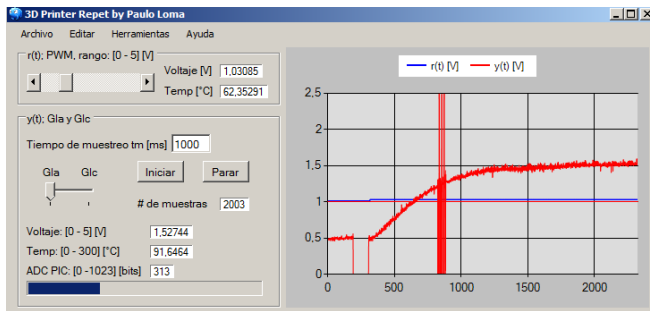


Figura 16: Respuesta G_p al escalón, muestreado con el GUI.

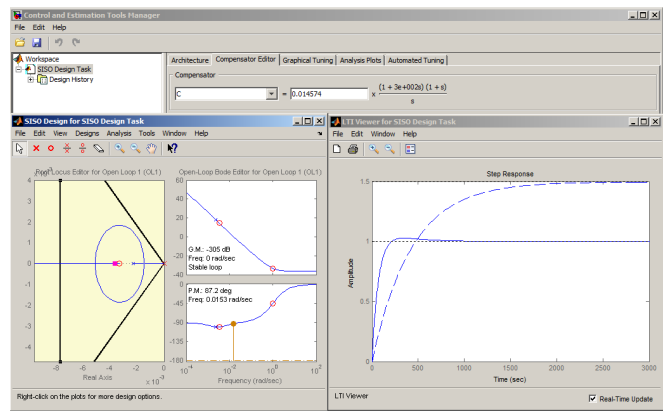


Figura 18: Controlador $G_c(s)$ con Sisotool.

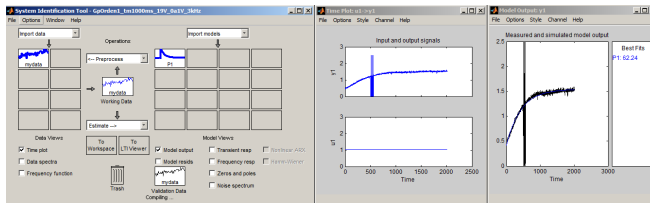


Figura 17: Identificación del modelo con Ident Matlab.

Que resulta en:

$$G_c(s) = 0,014574 \frac{(1+s)(1+300s)}{s} \quad (8)$$

$$K_d = 4,372 \quad K_p = 4,387 \quad K_i = 0,01457$$

$$M_p = 2,75\% \quad t_r = 127[s] \quad t_s = 507$$

3) Simulación

Se discretiza G_p , G_c y G_{lc} según el código Matlab 1, para obtener la figura 19.

```
1 s=tf('s');
2 Gp=1.4958/(1+429.15*s);
3 Kd=4.372; Kp=4.387; Ki=0.01457;
4 Gc=tf([Kd Kp Ki],[1 0]); % Gc=0.014574*(1+s)
   *(1+300*s)/s
```

4) Implementación

Para implementar el algoritmo PID 4, el MCU debe ejecutarlo en periodos de **Tiempo Discreto** $T = 1000[ms]$ donde el resultado se muestra en la figura 20.

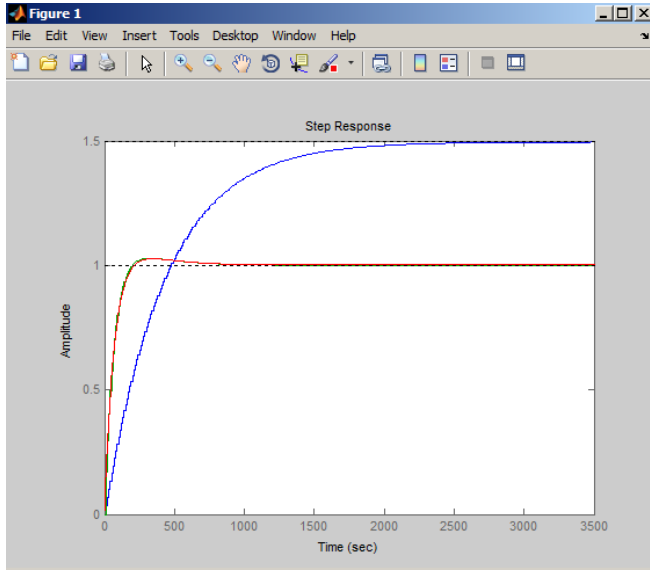
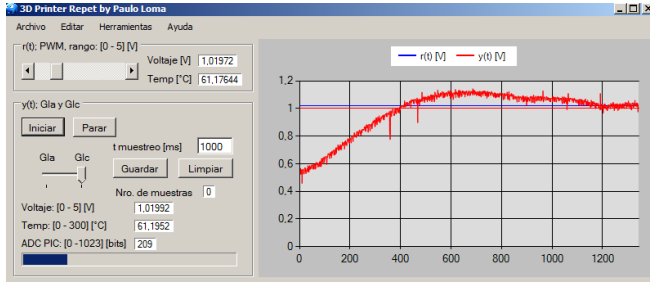


Figura 19: Simulación Matlab.

Figura 20: Respuesta al escalón del sistema a lazo cerrado G_{lc} .

IV. DESARROLLO DE SOFTWARE

El *Software* se realiza aplicando P00 a través de Visual Studio C# con diseño de interfaz WFA, la figura 21 muestra GUI completo con 5 secciones definidas:

- **Referencia $r(t)$** por PWM: rango de $0 - 5[V] = 0 - 300[^\circ C]$.
- **Salida $y(t)$** : elección $Gla - Glc$, Iniciar/Parar Muestreo, tiempo $t_m[ms]$; rango de salida por voltaje $0 - 5[V]$, temperatura $0 - 300[^\circ C]$ y $0 - 1023[dec]$.
- **Control Manual mPaP XYZE** con 2 subgrupos: número de Pulsos(distancia), periodo de Pulsos(velocidad).
- **Gcode**: cargar, instrucción por vez step, reset, timer, enviar una instrucción.
- **Monitoreo**: referencia $r(t)$ vs salida $y(t)$, a través de un *chart* o *plotter*.

Algoritmo 4: Control PID discreto.

Input: Periodo $T[ms]$, constantes K_{pz}, K_{iz}, K_{dz} , referencia $r(kT)$ y salida $y(kT)$

Output: Controlador $u(kT)$

def

```

 $e(kT) = e$     $e(kT - T) = e1$ ;
 $r(kT) = r$     $y(kT) = y$ ;
 $p(kT) = p$ ;
 $i(kT) = i$     $i(kT - T) = i1$ ;
 $d(kT) = d$ ;
 $u(kT) = u$ ;

```

Function control Glc_PID

```

 $e = r - y$  ; /* leer error actual          */
 $p = K_{pz} \cdot e$ ;
 $i = i1 + K_{iz} \cdot (e + e1)$ ;
 $d = K_{dz} \cdot (e - e1)$ ;
/* suma pid                                */
 $u = p + i + d$ ;
/* enviar dato de control "u" al          */
actuador(u);
/* error actual se guarda en error      */
anterior y valor integral actual en      */
valor integral anterior                  */
 $e1 = e$ ;
 $i1 = i$ ;
/* Periodo de muestreo                    */
delay = T[ms];

```

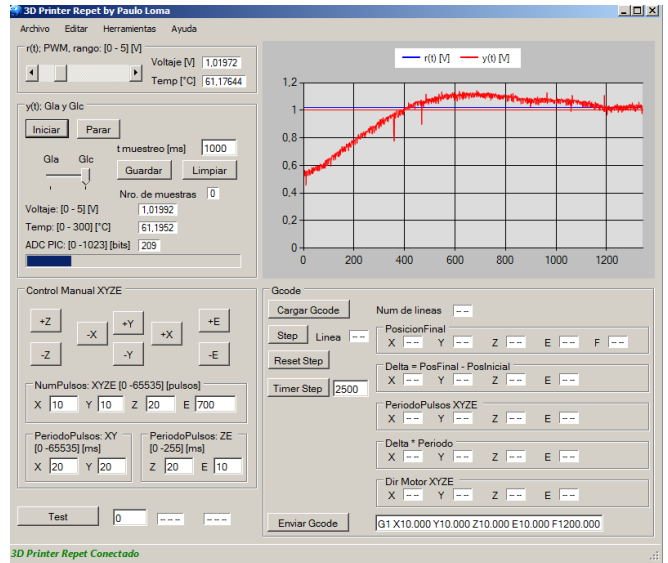


Figura 21: Operación general a través del GUI.

A. Descriptor Gcode

El Descriptor Gcode tiene la función de obtener los datos de posición XYZE y velocidad $F(Feedrate)$ de los mPaP, procesar dichos valores para enviar solamente

número de Pulsos XYZE y Periodo XZYE al *Firmware* para su ejecución.

El código ejemplo 2, indica el desarrollo de un rectángulo 2D sin el eje Z, donde el extrusor E deposita el material de acuerdo a las distancias XY [mm], G1 indica que la instrucción es de Movimiento Lineal Controlado, velocidad (*Feedrate*) $F 1200 \left[\frac{mm}{min} \right]$. Cada línea de instrucción refiere al *toolpath* necesario para completar el rectángulo 2D, figura 22.

```
G1 X0.000 Y0.000
G1 X10.000 E10.000 F1200.000
G1 Y5.000 E5.000 F1200.000
G1 X10.000 E10.000 F1200.000
G1 Y5.000 E5.000 F1200.000
```

Código 2: Ejemplo Gcode, rectángulo 2D de 10x5 mm.

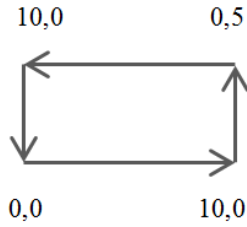


Figura 22: *Toolpath* del código 2.

Por tanto, el procedimiento de desarrollo del algoritmo descriptor consiste en:

- Guardar en *buffer* de memoria el *Documento.gcode* que contiene todas las instrucciones Gcode.
- Buscar la instrucción de Movimiento Lineal Controlado.
- Procesar el campo numérico para guardar en variables temporales XYZEF.
- Realizar el escalamiento de distancias XYZE, de acuerdo a la tabla 2.

	1[mm]	$10 \left[\frac{mm}{s} \right] \approx 20[ms]$
X	3,03[pulsos]	
Y	10[pulsos]	
Z	50[pulsos]	
E	70[pulsos]	
F		$2 \cdot 2 \cdot \frac{F \cdot 2}{60 \cdot 20} \cdot 20[ms]$

Tabla 2: Condiciones de relación (pulsos vs mm).

- Realizar el promedio de las distancias y periodos XYE para calcular la ecuación de relación; tomando como referencia al periodo del motor E, tabla 3.
- Sin embargo, no es necesario calcular la relación agregando el eje Z porque su movimiento solo se ejecuta cuando termina un capa.

	X	Y	E
[mm]	$d_{X1} \cdot 3,03 = d_{X2}$	$d_{Y1} \cdot 10 = d_{Y2}$	$d_{E1} \cdot 70 = d_{E2}$
[ms]	$T_{X1} \cdot 3,03 \cdot 7 = T_{X2}$	$T_{Y1} \cdot 7 = T_{Y2}$	$T_E = T_{ref}$

Tabla 3: Condiciones de relación (mm vs periodo).

- Relación XYE

$$d_{X2} \cdot T_{X2} = d_{Y2} \cdot T_{Y2} = d_{E2} \cdot T_E \quad (9)$$

$$T_{Y2} = \frac{d_{E2}}{d_{Y2}} \cdot T_E \quad (10)$$

$$T_{X2} = \frac{d_{E2}}{d_{X2}} \cdot T_E \quad (11)$$

V. RESULTADOS

Procedimiento de impresión:

- Implementado el siguiente modelo CAD, figura 23; y exportando el formato .stl
 - Interior: 7[mm]x12[mm]
 - Exterior: 15[mm]x20[mm]
 - Altura: 5[mm]
- Obteniendo el Documento.gcode con Slic3r y visualizando con Repetier-Host, figura 24.
- Importando el Documento.gcode con el botón Cargar Gcode e imprimiendo la pieza con el botón Step; usando el GUI del Proyecto, figura 21.
- Pieza final resultante, figura 25.
 - Interior: 7[mm]x13[mm]
 - Exterior: 116[mm]x21[mm]
 - Altura: 5[mm]

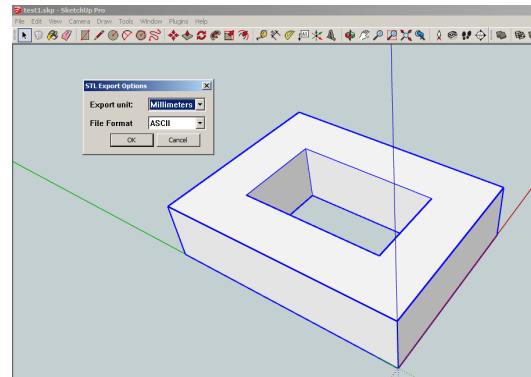


Figura 23: Modelo CAD de la pieza.

VI. CONCLUSIONES

- El extrusor electromecánico y todo el diseño electrónico fue implementado con *e-waste* y componentes adquiridos en tiendas locales, sección II.

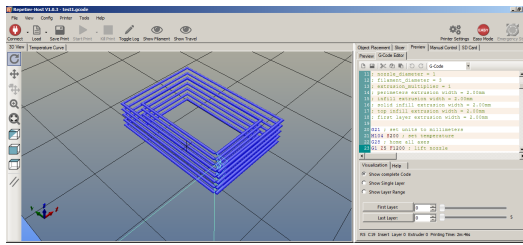


Figura 24: Generación y visualización Gcode.

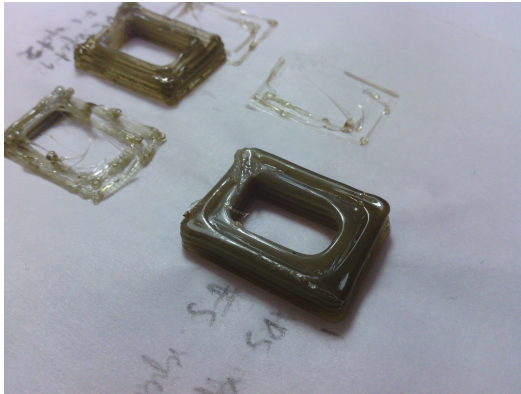


Figura 25: Pieza final impresa.

- La subsección B. demuestra la aplicación de material de impresión plástico PET de botellas desechables, alimentado al Licuefactor manualmente.
- El diseño del sistema electrónico y su implementación satisface todos los requerimientos de funcionamiento: control, comunicación y operación del prototipo, subsección F.
- El procedimiento de control PID se aplicó a través de las herramientas *Ident* y *Sisotool* de *Matlab*, la adquisición de datos de la planta G_p fue por medio del propio GUI del proyecto, y la implementación a través del MCU PIC18F4550, sección III.
- El desarrollo de *Firmware* sincroniza los mPaP para el correcto posicionamiento 3D, envía/recibe datos del sensor de temperatura por medio del protocolo USB para el correcto control PID, sección III.
- El desarrollo de *Software* aplicando P00, interactúa completamente con el *Firmware* para la ejecución de las instrucciones Gcode en el proceso de impresión 3D, sección IV.
- La baja resolución de las piezas impresas se debe a la limitación del Medio Paso en los mPaP; por tanto, se recomienda el cambio de diseño electrónico que implemente Micro Paso, que resultará en una Impresión 3D de calidad media/alta.
- La alimentación del material de impresión es manual y la cantidad de plástico procesado depende del volumen del Licuefactor, se recomienda un di-

seño nuevo de extrusor que contemple la alimentación continua.

REFERENCIAS

- [1] Alex B. OSA-RTOS. 2009. URL: <http://www.pic24.ru/doku.php/en/osa/ref/intro>.
- [2] Christian Baechler, Matthew DeVuono y Joshua Pearce. "Distributed recycling of waste polymer into RepRap feedstock". English. En: *Rapid Prototyping Journal* (2013), págs. 118-125. DOI: [dx.doi.org/10.1108/13552541311302978](https://doi.org/10.1108/13552541311302978). URL: [http://www.emeraldinsight.com/journals.htm?issn=1355-2546&volume=19&issue=2&articleid=17082798&articletitle=Distributed %20recycling %20of %20waste %20polymer%20into%20RepRap%20feedstock](http://www.emeraldinsight.com/journals.htm?issn=1355-2546&volume=19&issue=2&articleid=17082798&articletitle=Distributed%20recycling%20of%20waste%20polymer%20into%20RepRap%20feedstock).
- [3] David Bourell, Ming Leu y David Rosen. "Roadmap for Additive Manufacturing Identifying the Future of Freeform Processing". En: 2009. URL: <http://wohlersassociates.com/roadmap2009.pdf>.
- [4] Adrian Bowyer. *Granule Extruder*. URL: <http://reprap.org/wiki/GranuleExtruder>.
- [5] G.B. Braanker y col. *Developing a plastics recycling add-on for the RepRap 3D-printer*. Delft University of Technology, 2010. 46 págs. URL: <http://reprapdelft.files.wordpress.com/2010/04/reprap-granule-extruder-tudelft1.pdf>.
- [6] Enrique Canessa, Carlo Fonda y Marco Zennaro. "Perpetual Plastic Project". En: *Low-cost 3D Printing for Science, Education and Sustainable Development* (mayo de 2013), págs. 191-197. URL: http://sdu.ictp.it/3d/book/Low-cost_3D_printing_screen.pdf.
- [7] Eduardo Carletti. *Motores paso a paso Características básicas*. 2014. URL: http://robots-argentina.com.ar/MotorPP_basico.htm.
- [8] Miguel A. Pérez García y col. *Instrumentación Electrónica*. S.A. Ediciones Paraninfo, 2003. 862 págs. ISBN: 9788497321662. URL: <http://m.casadellibro.com/libro-instrumentacion-electronica-incluye-cd/9788497321662/936109>.
- [9] Jo Geraedts y col. "Three view on Additive Manufacturing: Business, Research and Education". En: *TMC2012*. Ed. por I. Horváth y col. Organizing Committee of TMCE 2012. Mayo de 2012. URL: http://www.researchgate.net/publication/235725722_Three_Views_on_Additive_Manufacturing_Business_Research_and_Education.

- [10] Ian Gibson, David Rosen y Brent Stucker. *Additive Manufacturing Technologies*. Springer US, 2010. DOI: [10.1007/978-1-4419-1120-9](https://doi.org/10.1007/978-1-4419-1120-9). URL: <http://link.springer.com/book/10.1007/978-1-4419-1120-9>.
- [11] Harold Giles, John Wagner y Eldridge Mount. *Extrusion: the definitive processing guide and handbook*. English. 2nd. William Andrew, sep. de 2014. 640 págs. ISBN: 9781437734829. URL: <http://store.elsevier.com/product.jsp?isbn=9781437734812&requestid=596110>.
- [12] Sherry Huss. *Ultimate Guide to 3D Printing*. Inf. téc. Makezine.com, 2013. URL: <http://www.makershed.com/products/make-ultimate-guide-to-3d-printing-2014>.
- [13] Pierre Lafleur y Bruno Vergnes. *Polymer Extrusion*. 1th Edition. Wiley-ISTE, 9 de mayo de 2014. URL: <http://www.wiley.com/WileyCDA/WileyTitle/productCd-1848216505.html>.
- [14] Keith Luker. "Hot-melt Extrusion: Pharmaceutical Applications". En: ed. por Dennis Douroumis. 1th Edition. A John Wiley, abr. de 2012. Cap. 1. URL: <http://www.wiley.com/WileyCDA/WileyTitle/productCd-1118307879.html>.
- [15] Joshua Pearce. *Waste plastic extruder: literature review*. 2014. URL: http://www.appropedia.org/Waste_Plastic_Extruder:_Literature_Review#Characterization_and_recovery_of_polymers_from_mobile_phone_scrap.
- [16] Dominik Proschek. "Extruder to Recycle Plastic Milk Bottles". Tesis de maestría. University of ULSTER. URL: http://reprap.org/mediawiki/images/9/90/Master_Thesis_Extruder_Unit.pdf.
- [17] Chris Rauwendaal y col. *Polymer Extrusion*. 5th Edition. Carl Hanser Verlag, 2014. URL: <http://www.hanser-elibrary.com/isbn/9781569905166>.
- [18] Bruno Vergnes, Michel Vincent y Jean-Marc Haudin. "Properties of Polymers". En: *Design of Extrusion Forming Tools*. Ed. por Olga Carneiro y Joao Nóbrega. 1th Edition. Smithers Rapra, 2012. Cap. 2. ISBN: 978-1-84735-519-5. URL: <http://info.smithersrapra.com/publishing/10B16-1004/design-of-extrusion-forming-tools/>.