



CARRERA DE ESPECIALIZACIÓN EN
SISTEMAS EMBEBIDOS

MEMORIA DEL TRABAJO FINAL

**Equipo dip coater para la creación de
películas delgadas**

Autor:
Ing. Martin Abel Gambarotta

Director:
Dr. Gastón Corthey (CONICET)

Jurados:
Alejandro Permingeat (FIUBA, DETECAP)
Diego Fernández (UBA)
Julián Iglesias (UTN)

Este trabajo fue realizado en la Ciudad de General San Martín, Buenos Aires,
entre marzo de 2020 y **julio** de 2022.



CARRERA DE ESPECIALIZACIÓN EN
SISTEMAS EMBEBIDOS

MEMORIA DEL TRABAJO FINAL

**Equipo dip coater para la creación de
películas delgadas**

Autor:
Ing. Martin Abel Gambarotta

Director:
Dr. Gastón Corthey (CONICET)

Jurados:
Alejandro Permingeat (FIUBA, DETECAP)
Diego Fernández (UBA)
Julián Iglesias (UTN)

Este trabajo fue realizado en la Ciudad de General San Martín, Buenos Aires,
entre marzo de 2020 y **abril** de 2022.

I

I

Resumen

La presente memoria describe el desarrollo y la implementación de un equipo dip coater utilizado en la fabricación de películas delgadas en el campo de estudio de las nanociencias. Se abarcarán aspectos de software, hardware y también de diseño y fabricación mecánica.

El equipo que surge de este proyecto será comercializado por TECSCI S.A.S. en el transcurso del año 2022. Todo el material relacionado estará disponible ya que la empresa adhiere a los principios del software y hardware libre.

Resumen

La presente memoria describe el desarrollo y la implementación de un equipo dip coater utilizado en la fabricación de películas delgadas en el campo de estudio de las nanociencias. Se abarcarán aspectos de software, hardware y también de diseño y fabricación mecánica.

El equipo que surge de este proyecto será comercializado por TECSCI S.A.S en el transcurso del año 2022. Todo el material relacionado estará disponible ya que la empresa adhiere a los principios del software y hardware libre.

Índice general

Resumen	I
1. Introducción general	1
1.1. Contexto	1
1.2. Técnicas de dip coating	2
1.3. Dip coaters en el mercado	4
1.4. Objetivos y alcance	5
1.4.1. Objetivos	5
1.4.2. Alcance	6
2. Introducción específica	7
2.1. Estudio preliminar	7
2.2. Circuitos integrados Trinamic	8
2.2.1. Driver TMC5130	9
2.3. Interfaz de usuario	11
2.4. Estructura mecánica	12
2.5. Sistema electrónico propuesto	13
2.6. Herramientas de desarrollo	15
3. Diseño e Implementación	17
3.1. Hardware	17
3.1.1. Diseño basado en módulos de hardware libre	17
3.1.2. Etapa de alimentación	17
3.1.3. Etapa de comunicación	18
3.1.4. Driver TMC5130	19
3.1.5. Diseño final	20
3.1.6. Fabricación	21
3.2. Firmware	22
3.2.1. Capas de abstracción	22
3.2.2. Módulos principales de software	23
Funcionamiento general	23
Control de movimientos	23
Consola de comandos	27
Pantalla táctil	29
Registro de variables ambientales	31
Parámetros de calibración	32
3.3. Estructura mecánica	33
3.3.1. Fabricación de piezas personalizadas a través de mecanizado CNC	33
Etapa CAD	33
Etapa CAM	34
3.3.2. Modelos 3D y real	34

Índice general

Resumen	I
1. Introducción general	1
1.1. Contexto	1
1.2. Técnicas de dip coating	2
1.3. Dip coaters en el mercado	3
1.4. Objetivos y alcance	5
1.4.1. Objetivos	5
1.4.2. Alcance	5
2. Introducción específica	7
2.1. Estudio preliminar	7
2.2. Circuitos integrados Trinamic	8
2.2.1. Driver TMC5130	9
2.3. Interfaz de usuario	11
2.4. Estructura mecánica	12
2.5. Sistema electrónico propuesto	13
2.6. Herramientas de desarrollo	15
3. Diseño e Implementación	17
3.1. Hardware	17
3.1.1. Diseño basado en módulos de hardware libre	17
3.1.2. Etapa de alimentación	17
3.1.3. Etapa de comunicación	18
3.1.4. Driver TMC5130	19
3.1.5. Diseño final	20
3.1.6. Fabricación	21
3.2. Firmware	22
3.2.1. Capas de abstracción	22
3.2.2. Módulos principales de software	23
Funcionamiento general	23
Control de movimientos	23
Consola de comandos	27
Pantalla táctil	29
Registro de variables ambientales	31
Parámetros de calibración	32
3.3. Estructura mecánica	33
3.3.1. Fabricación de piezas personalizadas a través de mecanizado CNC	33
Etapa CAD	33
Etapa CAM	34
3.3.2. Modelos 3D y real	34

4. Ensayos y resultados	39
4.1. Pruebas funcionales de hardware	39
4.1.1. Comunicación con driver TMC5130	39
4.2. Pruebas funcionales del firmware	41
4.2.1. Tiempo de ejecución de movimientos	41
4.2.2. Ejecución de comandos	42
4.3. Calibración del equipo	44
4.3.1. Desplazamiento lineal y micropasos	44
4.4. Caso de prueba	46
4.4.1. Prueba de campo con personal capacitado	46
4.5. Comparación con el estado del arte	49
5. Conclusiones	51
5.1. Resultados obtenidos	51
5.2. Próximos pasos	51
Bibliografía	53

4. Ensayos y resultados	39
4.1. Pruebas funcionales de hardware	39
4.1.1. Comunicación con driver TMC5130	39
4.2. Pruebas funcionales del firmware	41
4.2.1. Tiempo de ejecución de movimientos	41
4.2.2. Ejecución de comandos	42
4.3. Calibración del equipo	44
4.3.1. Desplazamiento lineal y micro pasos	44
4.4. Caso de prueba	47
4.4.1. Prueba de campo con personal capacitado	47
4.6. Comparación con el estado del arte	49
5. Conclusiones	51
5.1. Resultados obtenidos	51
5.2. Próximos pasos	51
Bibliografía	53

Índice de figuras

1.1. Centro Tecnológico FUNINTEC.	2
1.2. Proceso completo desarrollado por el equipo ¹ .	2
1.3. Films de dioxido de titanio TiO ₂ ² .	3
1.4. Equipo de la empresa Kibron.	4
1.5. Equipos de la empresa Biolin Scientific.	4
1.6. Equipo de la empresa Bungard.	5
2.1. Espesos vs velocidad ³ .	7
2.2. Placa de desarrollo Startrampe + placa de evaluación TMC5130 ⁴ .	9
2.3. Diagrama en bloques TMC5130 ⁵ .	9
2.4. Función stallguard2. ⁶	10
2.5. Función coolstep. ⁷	11
2.6. Guía Lineal IGUS. ⁸	13
2.7. Fresadora Fagor GVC 600. ⁹	14
2.8. Esquema de equipo propuesto.	15
3.1. Módulo NodeMCU + TMC5130-EVAL.	17
3.2. Módulo de entrada.	18
3.3. Conversor serie-USB.	18
3.4. Clock para el CI TMC5130.	19
3.5. CI TMC5130.	20
3.6. Modelo 3D Kicad.	20
3.7. Plaqueta electrónica final.	21
3.8. Capas de abstracción de software.	22
3.9. Software TMCL-IDE.	24
3.10. Configuración de funcionalidades stallguard2 y coolstep.	24
3.11. Configuración de rampa de seis puntos.	25
3.12. Comandos de movimientos.	28
3.13. Comandos de control.	29
3.14. Lectura de registros del driver TMC5130.	29
3.15. Datagrama desde microcontrolador hacia pantalla.	30
3.16. Datagrama desde pantalla hacia microcontrolador.	30
3.17. Secuencia de procesamientos de datos entrantes.	31
3.18. Pantalla de configuración de programa.	31
3.19. Módulo API BOSH.	32
3.20. Registro de datos en consola.	32
3.21. Unidades.	33
3.22. Pieza personalizada soporte de carro.	34
3.23. Piezas personalizada soporte de estructura superior.	35
3.24. Operaciones de mecanizado en software Bodcad.	36
3.25. Piezas fabricadas en centro de mecanizado.	37
3.26. Modelo 3D.	37
3.27. Primer prototipo dip coater TECSI.	38

Índice de figuras

1.1. Centro Tecnológico FUNINTEC.	2
1.2. Proceso completo desarrollado por el equipo ¹ .	2
1.3. Films de dioxido de titanio TiO ₂ ² .	3
1.4. Equipo de la empresa Kibron.	4
1.5. Equipos de la empresa Biolin Scientific.	4
1.6. Equipo de la empresa Bungard.	4
2.1. Espesos vs velocidad ³ .	7
2.2. Placa de desarrollo Startrampe + placa de evaluación TMC5130 ⁴ .	9
2.3. Diagrama en bloques TMC5130 ⁵ .	9
2.4. Función stallguard2. ⁶	10
2.5. Función coolstep. ⁷	11
2.6. Guía Lineal IGUS. ⁸	13
2.7. Fresadora Fagor GVC 600. ⁹	14
2.8. Esquema de equipo propuesto.	15
3.1. Módulo NodeMCU + TMC5130-EVAL.	17
3.2. Módulo de entrada.	18
3.3. Conversor serie-USB.	18
3.4. Clock para el CI TMC5130.	19
3.5. CI TMC5130.	20
3.6. Modelo 3D Kicad.	20
3.7. Plaqueta electrónica final.	21
3.8. Capas de abstracción de software.	22
3.9. Software TMCL-IDE.	24
3.10. Configuración de funcionalidades stallguard2 y coolstep.	25
3.11. Configuración de rampa de seis puntos.	25
3.12. Comandos de movimientos.	28
3.13. Comandos de control.	29
3.14. Lectura de registros del driver TMC5130.	29
3.15. Datagrama desde microcontrolador hacia pantalla.	30
3.16. Datagrama desde pantalla hacia microcontrolador.	30
3.17. Secuencia de procesamientos de datos entrantes.	31
3.18. Pantalla de configuración de programa.	31
3.19. Módulo API BOSH.	32
3.20. Registro de datos en consola.	32
3.21. Unidades.	33
3.22. Pieza personalizada soporte de carro.	34
3.23. Piezas personalizada soporte de estructura superior.	34
3.24. Operaciones de mecanizado en software Bodcad.	35
3.25. Piezas fabricadas en centro de mecanizado.	36
3.26. Modelo 3D.	37
3.27. Primer prototipo dip coater TECSI.	37

Índice de tablas

1.1. Dip coaters en el mercado	5
2.1. Comparación Stone	12
4.1. Ensayo de tiempos en desplazamientos	42
4.2. Ensayo de desplazamiento descendente	46
4.3. Ensayo de desplazamiento ascendente	46
4.4. Dip coaters en el mercado	49

Índice de tablas

1.1. Dip coaters en el mercado	5
2.1. Comparación Stone	12
4.1. Ensayo de tiempo en desplazamientos	42
4.2. Ensayo de desplazamiento	46
4.3. Ensayo de desplazamiento	46
4.4. Dip coaters en el mercado	49

- Láser de corte.
- Cabina de *blasting*

En la figura 1.1 se observan los equipos mencionados.



FIGURA 1.1. Centro Tecnológico FUNINTEC.

La empresa también cuenta con un laboratorio de electrónica con equipos para diseño y prototipado electrónico.

1.2. Técnicas de dip coating

En los laboratorios de investigación aplicados en nanotecnologías existen diferentes equipos para la fabricación de películas delgadas o *thin films*. Las mismas consisten en capas de material de espesores variables, que comúnmente van desde las centenas de nanómetros hasta las decenas de micrómetros y se depositan sobre diferentes superficies.

Dip coating es una técnica que se emplea tanto en áreas de I+D (investigación y desarrollo) en la industria, como en la investigación científica en el campo de las nanociencias, se basa en la inmersión y extracción controlada de un sustrato en una solución química bajo estudio. En la figura 1.2 se observa una ejecución completa del movimiento desarrollado que consta de los siguientes pasos:

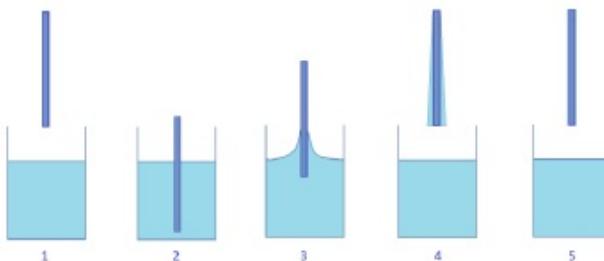


FIGURA 1.2. Proceso completo desarrollado por el equipo ¹.

¹Imagen tomada de [5]

- Láser de corte.
- Cabina de *blasting*

En la siguiente figura 1.1 se observan los equipos mencionados.



FIGURA 1.1. Centro Tecnológico FUNINTEC.

La empresa también cuenta con un laboratorio de electrónica con equipos para diseño y prototipado electrónico.

1.2. Técnicas de dip coating

En los laboratorios de investigación aplicados en nanotecnologías existen diferentes equipos para la fabricación de películas delgadas o *thin films*. Las mismas consisten en capas de material de espesores variables, que comúnmente van desde las centenas de nanómetros hasta las decenas de micrómetros y se depositan sobre diferentes superficies.

Dip coating es una técnica que se emplea tanto en áreas de I+D (investigación y desarrollo) en la industria, como en la investigación científica en el campo de las nanociencias, se basa en la inmersión y extracción controlada de un sustrato en una solución química bajo estudio. En la figura 1.2 se observa una ejecución completa del movimiento desarrollado:

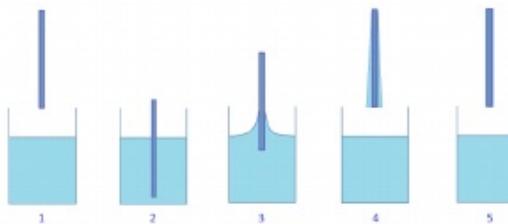


FIGURA 1.2. Proceso completo desarrollado por el equipo ¹.

1. La muestra desciende a velocidad controlada.
2. La muestra queda sumergida un tiempo establecido por el usuario.

¹Imagen tomada de [5]

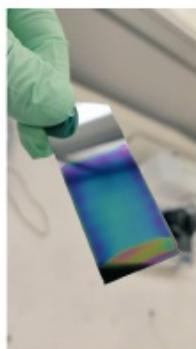
1.2. Técnicas de dip coating

3

1. La muestra desciende a velocidad controlada.
2. La muestra queda sumergida un tiempo establecido por el usuario.
3. La muestra asciende a velocidad controlada, este es el punto más crítico del experimento, en donde el material queda adherido a la muestra. Se estudiarán en el capítulo 3 dos modelos matemáticos que explican este fenómeno y se dará una noción mas detallada de las velocidades que caracterizan al proceso.
4. Se extrae toda la muestra.
5. El usuario puede tener interés en volver a repetir el proceso un tiempo después.

La principal característica del equipo es brindarle al usuario la posibilidad de controlar la velocidad y aceleración de inmersión de la muestra, el tiempo de espera que la muestra queda sumergida y la extracción, teniendo la posibilidad de repetir el ciclo según se deseé.

Como ejemplo de los resultados que se obtienen aplicando esta técnica se observan en la figura 1.3 films de dioxido de titanio TiO_2 . En la imagen A el film se preparó sobre un wafer de silicio y en la imagen B sobre un portaobjeto de vidrio.



(A) Film sobre wafer de silicio.



(B) Film sobre portaobjeto.

FIGURA 1.3. Films de dioxido de titanio TiO_2 ².

Cabe destacar que los espesores de capa logrados en este experimento fueron de entre 180 nm y 200 nm, con velocidades de inmersión y extracción configuradas en 180 mm/min. Valores de velocidad similares deberán estar contemplados en el desarrollo del presente equipo dip coater.

Luego de un proceso dip coating, dependiendo del tipo de muestra que se genere, es necesario realizar tratamientos térmicos para finalizar el proceso. Los mismos se realizan con otro tipo de equipos y no forman parte del trabajo realizado.

²Imagen tomada en los laboratorios del Instituto de Nanosistemas de la UNSAM.

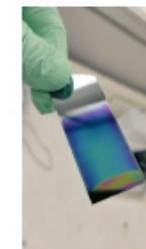
1.3. Dip coaters en el mercado

3

3. La muestra asciende a velocidad controlada, este es el punto más crítico del experimento, en donde el material queda adherido a la muestra. Se estudiarán en el capítulo 3 dos modelos matemáticos que explican este fenómeno y se dará una noción mas detallada de las velocidades que caracterizan al proceso.
4. Se extrae toda la muestra.
5. El usuario puede tener interés en volver a repetir el proceso un tiempo después.

La principal característica del equipo es brindarle al usuario la posibilidad de controlar la velocidad y aceleración de inmersión de la muestra, el tiempo de espera que la muestra queda sumergida y la extracción, teniendo la posibilidad de repetir el ciclo según se deseé.

Como ejemplo de los resultados que se obtienen aplicando esta técnica se observan en la figura 1.3 films de dioxido de titanio TiO_2 . En la imagen A el film se preparó sobre un wafer de silicio y en la imagen B sobre un portaobjeto de vidrio.



(A) Film sobre wafer de silicio.



(B) Film sobre portaobjeto.

FIGURA 1.3. Films de dioxido de titanio TiO_2 ².

Cabe destacar que los espesores de capa logrados en este experimento fueron de entre 180 nm y 200 nm, con velocidades de inmersión y extracción configuradas en 180 mm/min. Valores de velocidad similares deberán estar contemplados en el desarrollo del presente equipo dip coater.

Luego de un proceso dip coating, dependiendo del tipo de muestra que se genere, es necesario realizar tratamientos térmicos para finalizar el proceso. Los mismos se realizan con otro tipo de equipos y no forman parte del trabajo realizado.

1.3. Dip coaters en el mercado

Existen diferentes fabricantes a nivel internacional que comercializan este tipo de equipos, pero ninguno a nivel local. Se presentan a continuación algunos productos de las empresas internacionales con mayor reconocimiento en el ámbito de las nanociencias.

Se puede observar en la figura 1.4 el equipo de la empresa Kibron [6].

²Imagen tomada en los laboratorios del Instituto de Nanosistemas de la UNSAM.

1.3. Dip coaters en el mercado

Existen diferentes fabricantes a nivel internacional que comercializan este tipo de equipos, pero ninguno a nivel local. Se presentan a continuación algunos productos de las empresas internacionales con mayor reconocimiento en el ámbito de las nanociencias.

Se puede observar en la figura 1.4 el equipo de la empresa Kibron [6].



FIGURA 1.4. Equipo de la empresa Kibron.

En la figura 1.5 se observan los equipos de la empresa Biolin Scientific [7], un equipo simple y otro con mayor funcionalidad. Si bien ambos controlan con exactitud la velocidad de inmersión y extracción, el último agrega una funcionalidad extra que, a través de una rotación en la base, da la posibilidad de cambiar de manera automática y secuencial las soluciones donde se realizan las inmersiones. Ambos equipos necesitan estar conectados a una computadora corriendo un software para poder ser accionados.



(A) Equipo simple.



(B) Equipo avanzado.

FIGURA 1.5. Equipos de la empresa Biolin Scientific.

Por último se presenta el equipo de la empresa Bungard [8], que puede observarse en la figura 1.6. Este equipo a diferencia de los otros cuenta con un display LCD y botonera, lo que permite al usuario realizar una configuración a pie de máquina.



FIGURA 1.4. Equipo de la empresa Kibron.

En la figura 1.5 se observan los equipos de la empresa Biolin Scientific [7], un equipo simple y otro con mayor funcionalidad. Si bien ambos controlan con exactitud la velocidad de inmersión y extracción, el último agrega una funcionalidad extra, que a través de una rotación en la base, da la posibilidad de cambiar de manera automática y secuencial las soluciones donde se realizan las inmersiones. Ambos equipos necesitan estar conectados a una computadora corriendo un software para poder ser accionados.



(A) Equipo simple.



(B) Equipo avanzado.

FIGURA 1.5. Equipos de la empresa Biolin Scientific.

Por último se presenta el equipo de la empresa Bungard [8], que puede observarse en la figura 1.6. Este equipo a diferencia de los otros cuenta con un display LCD y botonera, lo que permite al usuario realizar una configuración a pie de máquina.



FIGURA 1.6. Equipo de la empresa Bungard.

A continuación se presenta la tabla 1.1 en donde se comparan las especificaciones técnicas que los caracterizan.

1.4. Objetivos y alcance

5



FIGURA 1.6. Equipo de la empresa Bungard.

A continuación se presenta la tabla 1.1 en donde se comparan las especificaciones técnicas que los caracterizan.

TABLA 1.1. Especificaciones técnicas de otros equipos.

Equipo	Recorrido (mm)	Velocidad (mm/min)	Aceleración (m/min ²)	Interface	Open Source Hardware
Bio Single	300	1 - 1000	no	PC	no
Bio Multiplie	70	0.1 - 108	no	PC	no
Kibron LayerX	134	0.06 - 300	no	PC	no
Bungard	600	30 - 10000	no	Display LCD	no
Ossila [9]	100	0.6 - 3000	no	PC	no
Holmarc [10]	100	1.08 - 540	no	PC	no

Del análisis de la tabla se pueden extraer algunas conclusiones: ninguno de los equipos permite al usuario tener control sobre la aceleración en los movimientos de inmersión y extracción de muestra, la mayoría de los equipos depende de una comunicación USB-SERIAL con una computadora, ninguna de las empresas adhiere a los principios del software y hardware libre.

También es importante aclarar que las investigaciones realizadas indican que el rango de velocidades más utilizado para la generación de *films* es el de 1 a 200 mm/min. Se destaca que la mayoría de los equipos presentados cumplen con dicha indicación.

1.4. Objetivos y alcance

1.4.1. Objetivos

El objetivo de este trabajo fue diseñar y fabricar el primer equipo comercial de la empresa TECSCI, con la perspectiva a futuro de ampliar la gama ofrecida de equipos de laboratorio para la investigación científica.

También es parte de los objetivos fundamentales que el equipo desarrollado incorpore mejoras respecto a sus competidores. Se planteará en los siguientes capítulos un estudio sobre el control de movimientos elegido y se presentará un sistema moderno de configuración de equipo.

1.4. Objetivos y alcance

5

TABLA 1.1. Especificaciones técnicas de otros equipos.

Equipo	Recorrido	Velocidad (mm/min)	Aceleración (m/min ²)	Interface
Bio Single Vessel M	300 mm	1 - 1000	no	PC
Bio Multiplie Vessel	70 mm	0.1 - 108	no	PC
Kibron LayerX	134 mm	0.06 - 300	no	PC
Bungard	600 mm	30 - 10000	no	Display LCD
Ossila [9]	100 mm	0.6 - 3000	no	PC
Holmarc [10]	100 mm	1.08 - 540	no	PC

Se realizaron consultas a investigadores científicos, los cuales indicaron que el rango de velocidades más utilizado para la generación de *films* es el de [1 a 200 mm/min]. Se destaca que la mayoría de los equipos presentados cumplen con dicha indicación.

Del análisis de la tabla se pueden extraer algunas conclusiones: ninguno de los equipos permite al usuario tener control sobre la aceleración en los movimientos de inmersión y extracción de muestra, la mayoría de los equipos depende de una comunicación USB-SERIAL con una computadora, ninguna de las empresas adhiere a los principios del software y hardware libre.

1.4. Objetivos y alcance

1.4.1. Objetivos

El objetivo de este trabajo fue diseñar y fabricar el primer equipo comercial de la empresa TECSCI, con la perspectiva a futuro de ampliar la gama ofrecida de equipos de laboratorio para la investigación científica.

También es parte de los objetivos fundamentales que el equipo desarrollado incorpore mejoras respecto a sus competidores. Se planteará en los siguientes capítulos un estudio sobre el control de movimientos elegido y se presentará un sistema moderno de configuración de equipo.

1.4.2. Alcance

El presente trabajo abarca la presentación de un MVP (Producto Mínimo Viable) de equipo dip coater.

El trabajo realizado incluye:

- Driver de motor provisto por el fabricante TRINAMIC [11].
- Diseño de hardware con software de diseño KICAD [12].
- Fabricación de placa electrónica y montaje de componentes.
- Diseño y fabricación de partes mecánicas a través del mecanizado de aluminio.
- Incorporación de pantalla táctil para configuración y uso del equipo.

El trabajo no incluye:

1.4.2. Alcance

El presente trabajo abarca la presentación de un MVP (Producto Mínimo Viable) de equipo dip coater.

El trabajo realizado incluye:

- Driver de motor provisto por el fabricante TRINAMIC [9].
- Diseño de hardware con software de diseño KICAD [10].
- Fabricación de placa electrónica y montaje de componentes.
- Diseño y fabricación de partes mecánicas a través del mecanizado de aluminio.
- Incorporación de pantalla táctil para configuración y uso del equipo.

El trabajo no incluye:

- Desarrollo de hardware con fuente de alimentación incorporada.
- Programación de la interfaz gráfica con el software de diseño provisto por el fabricante de la pantalla.
- Control del entorno con cámara de humedad.

- Desarrollo de hardware con fuente de alimentación incorporada.
- Programación de la interfaz gráfica con el software de diseño provisto por el fabricante de la pantalla.
- Control del entorno con cámara de humedad.

Capítulo 2

Introducción específica

En el presente capítulo se introducen los módulos principales del equipo dip coater fabricado.

2.1. Estudio preliminar

Para entender la relación entre la velocidad de extracción y el espesor de material depositado se tuvo en consideración la siguiente publicación (*Preparation of Sol-Gel Films by Dip-Coating*) [11], que describe la técnica dip coating como un proceso dinámico, complejo y difícil de modelar, debido a los gradientes de concentración y viscosidad generados por evaporación de la solución.

La publicación se basa entonces en un estudio semi-experimental sobre varias soluciones químicas para predecir el espesor final de la película. Tiene en cuenta dos modelos matemáticos, un modelo de capilaridad asociado a extracciones en velocidades bajas y otro modelo de evaporación asociado a velocidades altas respecto al rango de estudio.

Se observa en la figura 2.1 la variación de los espesores fabricados respecto a las velocidades utilizadas, también se puede observar la relación entre los diferentes modelos aplicados.

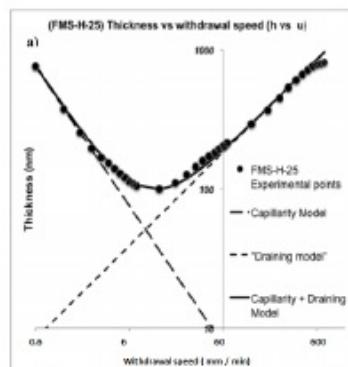


FIGURA 2.1. Espesor vs velocidad¹.

¹Imagen tomada de [11].

Capítulo 2

Introducción específica

En el presente capítulo se introducen los módulos principales del equipo dip coater fabricado.

2.1. Estudio preliminar

Para entender la relación entre la velocidad de extracción y el espesor de material depositado se tuvo en consideración la siguiente publicación (*Preparation of Sol-Gel Films by Dip-Coating*) [13], que describe la técnica dip coating como un proceso dinámico, complejo y difícil de modelar, debido a los gradientes de concentración y viscosidad generados por evaporación de la solución.

La publicación se basa entonces en un estudio semi-experimental sobre varias soluciones químicas para predecir el espesor final de la película. Tiene en cuenta dos modelos matemáticos, un modelo de capilaridad asociado a extracciones en velocidades bajas y otro modelo de evaporación asociado a velocidades altas respecto al rango de estudio.

Se observa en la figura 2.1 la variación de los espesores fabricados respecto a las velocidades utilizadas, también se puede observar la relación entre los diferentes modelos aplicados.

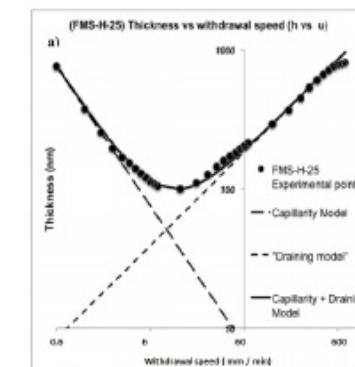


FIGURA 2.1. Espesor vs velocidad¹.

¹Imagen tomada de [13].

Los resultados del experimento concluyen en que existe linealidad en la relación de espesor respecto a la velocidad de extracción entre 60 $\frac{\text{mm}}{\text{min}}$ y 600 $\frac{\text{mm}}{\text{min}}$. También demuestra que en dicho rango de velocidades el fenómeno se explica por el modelo de evaporación.

Se desprende de este análisis la importancia de los siguientes requerimientos funcionales definidos por el cliente:

- El sistema debe contar con un rango de velocidades de desplazamiento de muestra entre [1- 1000 $\frac{\text{mm}}{\text{min}}$].
- El sistema debe contar con un rango de aceleraciones de desplazamiento de muestra entre [1000 - 15000 $\frac{\text{m}}{\text{min}^2}$].

Cabe destacar que todos los experimentos en la publicación fueron realizados a velocidad constante. De las reuniones con el cliente y del interés de trabajar en la frontera de la ciencia surgió la necesidad de poder darle al usuario la posibilidad de realizar experimentos a velocidad y aceleración controlada. Esto último es una cualidad que diferencia a este equipo respecto de todos los equipos comerciales analizados en la sección 1.3.

2.2. Circuitos integrados Trinamic

De reuniones con el cliente en donde se **remarcó** la importancia de trabajar con un control preciso de motor y en base a experiencias de uso de equipos con **tecnologías similares**, surge el interés de trabajar con un fabricante de driver específico. Se definen entonces los siguientes requerimientos:

- El equipo deberá contar con un motor paso a paso Nema 17 [12] para realizar los movimientos.
- Se utilizará un driver de motor de la marca Trinamic Motion Control.

Trinamic [9] se especializa en la fabricación de CI (Circuitos Integrados) para el control de diferentes tipos de motores, su tecnología se basa en convertir **señales** digitales en movimientos controlados. Tiene una amplia experiencia en la **industria** del control de motores y sus CI son utilizados en una gran variedad de productos. Actualmente fue adquirida por la compañía Analog Devices [13].

Cabe destacar que los integrados fabricados por la empresa Trinamic se utilizan en diversas aplicaciones en donde la precisión es importante, como por ejemplo: impresión 3D, automatización industrial, robótica y equipos de laboratorio médico entre otras. Cuenta con una amplia gama de productos que se diferencian principalmente según el tipo de motor que se quiera accionar. Luego de estudiar las diferentes alternativas ofrecidas se eligió trabajar con el driver TMC5130 [14].

Todos los CI requieren una configuración inicial de parámetros que depende del tipo de motor y de la carga asociada al mismo. Para encontrarla la empresa ofrece el software TMCL-IDE y diferentes placas de desarrollo para trabajar sobre los diferentes drivers. La placa de desarrollo que corresponde al integrado seleccionado es la *TMC5130-Eval Evaluation Board* [15].

Los resultados del experimento concluyen en que existe linealidad en la relación de espesor respecto a la velocidad de extracción entre 60 $\frac{\text{mm}}{\text{min}}$ y 600 $\frac{\text{mm}}{\text{min}}$. También demuestra que en dicho rango de velocidades el fenómeno se explica por el modelo de evaporación.

Se desprende de este análisis la importancia de los siguientes requerimientos funcionales definidos por el cliente:

- El sistema debe contar con un rango de velocidades de desplazamiento de muestra entre [1- 1000 $\frac{\text{mm}}{\text{min}}$].
- El sistema debe contar con un rango de aceleraciones de desplazamiento de muestra entre [1000 - 15000 $\frac{\text{m}}{\text{min}^2}$].

Cabe destacar que todos los experimentos en la publicación fueron realizados a velocidad constante. De las reuniones con el cliente y del interés de trabajar en la frontera de la ciencia surgió la necesidad de poder darle al usuario la posibilidad de realizar experimentos a velocidad y aceleración controlada. Esto último es una cualidad que diferencia a este equipo respecto de todos los equipos comerciales analizados en la sección 1.3.

2.2. Circuitos integrados Trinamic

De reuniones con el cliente en donde se **remarcó** la importancia de trabajar con un control preciso de motor y en base a experiencias de uso de equipos con **tecnologías similares**, surge el interés de trabajar con un fabricante de driver específico. Se definen entonces los siguientes requerimientos:

- El equipo deberá contar con un motor paso a paso Nema 17 [14] para realizar los movimientos.
- Se utilizará un driver de motor de la marca Trinamic Motion Control.

Trinamic [11] se especializa en la fabricación de CI (Circuitos Integrados) para el control de diferentes tipos de motores, su tecnología se basa en convertir **señales** digitales en movimientos controlados. Tiene una amplia experiencia en la **industria** del control de motores y sus CI son utilizados en una gran variedad de productos. Actualmente fue adquirida por la compañía Analog Devices [15].

Cabe destacar que los integrados fabricados por la empresa Trinamic se utilizan en diversas aplicaciones en donde la precisión es importante, como por ejemplo: impresión 3D, automatización industrial, robótica y equipos de laboratorio médico entre otras. Cuenta con una amplia gama de productos que se diferencian principalmente según el tipo de motor que se quiera accionar. Luego de estudiar las diferentes alternativas ofrecidas se eligió trabajar con el driver TMC5130 [16].

Todos los CI requieren una configuración inicial de parámetros que depende del tipo de motor y de la carga asociada al mismo. Para encontrarla la empresa ofrece el software TMCL-IDE y diferentes placas de desarrollo para trabajar sobre los diferentes drivers. La placa de desarrollo que corresponde al integrado seleccionado es la *TMC5130-Eval Evaluation Board* [17].

2.2. Circuitos integrados Trinamic

9

El TMCL-IDE se ejecuta sobre una placa de desarrollo general compatible con diferentes kits de evaluación. En la figura 2.2 se observa a izquierda la placa Startrampe que se conecta entre la computadora y la placa de evaluación TMC5130-Eval que se observa a derecha.

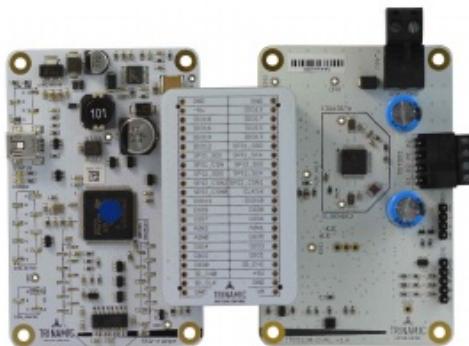


FIGURA 2.2. Placa de desarrollo Startrampe + placa de evaluación TMC5130².

2.2.1. Driver TMC5130

El driver TMC5130 permite operar motores bipolares de dos fases comúnmente conocidos como motores paso a paso. El CI incorpora una etapa de potencia con tecnología *MOSFET* (*Metal Oxide Semiconductor Field Effect Transistor*) que permite manejar corrientes de hasta dos amperios por fase. Se observa en la figura 2.3 el diagrama en bloque del CI.

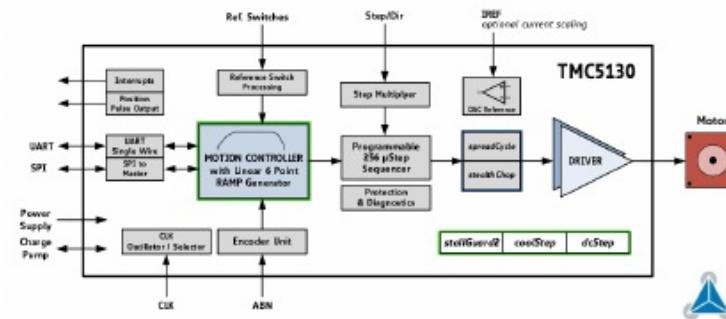


FIGURA 2.3. Diagrama en bloques TMC5130³.

La comunicación con el CI se puede establecer a través del protocolo serie o *SPI* (*Serial Peripheral Interface*), para el desarrollo de este trabajo se utilizó el protocolo de comunicación SPI.

²Imagen tomada de [9].

2.2. Circuitos integrados Trinamic

9

El TMCL-IDE se ejecuta sobre una placa de desarrollo general compatible con diferentes kits de evaluación. En la figura 2.2 se observa a izquierda la placa Startrampe que se conecta entre la computadora y la placa de evaluación TMC5130-Eval que se observa a derecha.

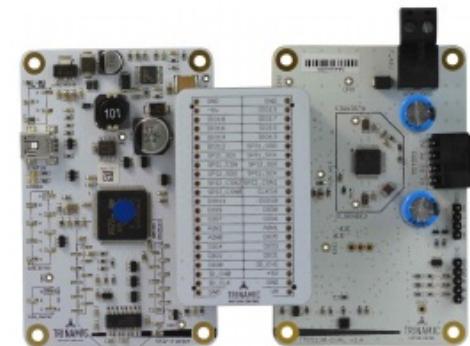


FIGURA 2.2. Placa de desarrollo Startrampe + placa de evaluación TMC5130².

2.2.1. Driver TMC5130

El driver TMC5130 permite operar motores bipolares de dos fases comúnmente conocidos como motores paso a paso. El CI incorpora una etapa de potencia con tecnología *MOSFET* (*Metal Oxide Semiconductor Field Effect Transistor*) que permite manejar corrientes de hasta dos amperios por fase. Se observa en la figura 2.3 el diagrama en bloque del CI.

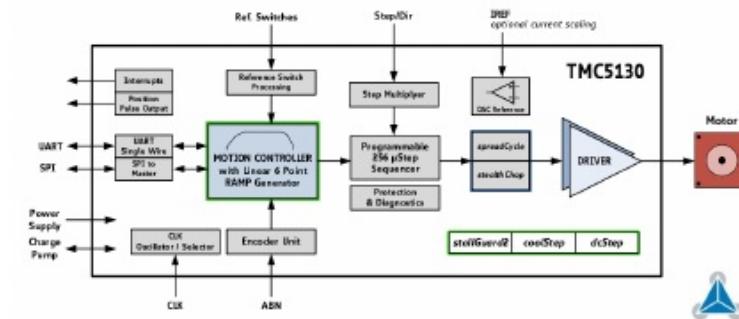


FIGURA 2.3. Diagrama en bloques TMC5130³.

La comunicación con el CI se puede establecer a través del protocolo serie o *SPI* (*Serial Peripheral Interface*), para el desarrollo de este trabajo se utilizó el protocolo de comunicación SPI.

³Imagen tomada de [11].

Los pasos que definen a este tipo de motores están relacionados con las fases y con la cantidad de dientes que tienen en su rotor y estator. Un paso es el movimiento mínimo que el motor puede hacer. Un motor pasa a paso, como su nombre lo indica, realiza movimientos a través de pasos sucesivos. Por ejemplo, es común contar con algún motor en donde la especificación indica que el paso es de 1.8° , esto significa que por cada vuelta de motor 360° , el mismo realizará 200 pasos.

Una funcionalidad que incorpora este CI es incrementar la cantidad de pasos, el fabricante los denomina micropasos. El driver puede generar hasta un máximo de 256 micropasos por cada paso del motor. Siguiendo con el ejemplo recién presentado, para un motor de paso 1.8° se tendrán en total 51200 micropasos, como se observa en la ecuación 2.1.

$$(360/1,8) * 256 = 51200 \text{ micropasos por revolución} \quad (2.1)$$

Una característica importante a destacar es la posibilidad de programar la cantidad de pasos que da el motor. El CI cuenta con el registro XACTUAL que contiene la cantidad de pasos absolutos desde una referencia inicial. También cuenta con el registro XTARGET que contiene una posición objetivo, cuando se escribe este registro el CI se acciona hasta lograr que XACTUAL = XTARGET.

Otra funcionalidad que se utilizó fue stallguard2, una función que mide la fuerza contraelectromotriz generada en las bobinas del motor por cambios de carga en el eje. En la figura 2.4 se observa que el valor del registro stallguard2 se decremente linealmente a medida que la carga aumenta. Cuando se aplica una fuerza contraria al movimiento programado o el recorrido del carro llega a un límite mecánico, la fuerza contraelectromotriz aumenta.

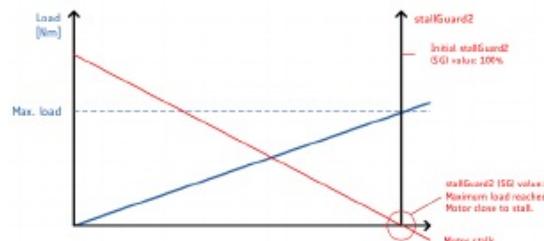


FIGURA 2.4. Función stallguard2.⁴

En el capítulo 3 se estudiará el valor del registro stallguard2 configurado. Cada vez que el equipo se enciende se realiza un movimiento hacia un extremo del recorrido para buscar el cero de máquina. Se utiliza esta medida para encontrar un límite mecánico del sistema y realizar un posicionamiento inicial. El uso de esta funcionalidad evita la incorporación de finales de carrera electromecánicos.

También se utilizó coolstep, una función que a través de mediciones de carga en el eje del motor adapta automáticamente la corriente suministrada hacia las bobinas, aumentando la eficiencia energética como puede observarse en la figura 2.5.

⁴Imagen tomada de [9].

Los pasos que definen a este tipo de motores están relacionados con las fases y con la cantidad de dientes que tienen en su rotor y estator. Un paso es el movimiento mínimo que el motor puede hacer. Un motor pasa a paso, como su nombre lo indica, realiza movimientos a través de pasos sucesivos. Por ejemplo, es común contar con algún motor en donde la especificación indica que el paso es de 1.8° , esto significa que por cada vuelta de motor 360° , el mismo realizará 200 pasos.

Una funcionalidad que incorpora este CI es incrementar la cantidad de pasos, el fabricante los denomina micropasos. El driver puede generar hasta un máximo de 256 micropasos por cada paso del motor. Siguiendo con el ejemplo recién presentado, para un motor de paso 1.8° se tendrán en total 51200 micropasos, como se observa en la ecuación 2.1.

$$(360/1,8) * 256 = 51200 \text{ micropasos por revolución} \quad (2.1)$$

Una característica importante a destacar es la posibilidad de programar la cantidad de pasos que da el motor. El CI cuenta con el registro XACTUAL que tiene contiene la cantidad de pasos absolutos desde una referencia inicial. También cuenta con el registro XTARGET que tiene una posición objetivo, cuando se escribe este registro el CI se accionara hasta lograr que XACTUAL = XTARGET.

Otra funcionalidad que se utilizó fue stallguard2, una función que mide la fuerza contraelectromotriz generada en las bobinas del motor por cambios de carga en el eje. En la figura 2.4 se observa que el valor del registro stallguard2 se decremente linealmente a medida que la carga aumenta. Cuando se aplica una fuerza contraria al movimiento programado o el recorrido del carro llega a un límite mecánico, la fuerza contraelectromotriz aumenta.

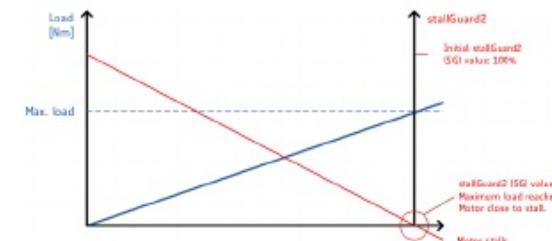


FIGURA 2.4. Función stallguard2.⁴

En el capítulo 3 se estudiará el valor del registro stallguard2 configurado. Cada vez que el equipo se enciende se realiza un movimiento hacia un extremo del recorrido para buscar el cero de máquina. Se utiliza esta medida para encontrar un límite mecánico del sistema y realizar un posicionamiento inicial. El uso de esta funcionalidad evita la incorporación de finales de carrera electromecánicos.

También se utilizó coolstep, una función que a través de mediciones de carga en el eje del motor adapta automáticamente la corriente suministrada hacia las bobinas, aumentando la eficiencia energética como puede observarse en la figura 2.5.

⁴Imagen tomada de [11].

2.3. Interfaz de usuario

11

El efecto final es reducir la energía suministrada según la hoja de datos [14] hasta un 75 %. Esto aplica incluso en equipos donde la carga es constante, como es el caso del dip coater, ya que la carga variable representada por un wafer de silicio o un portaobjeto es completamente despreciable.

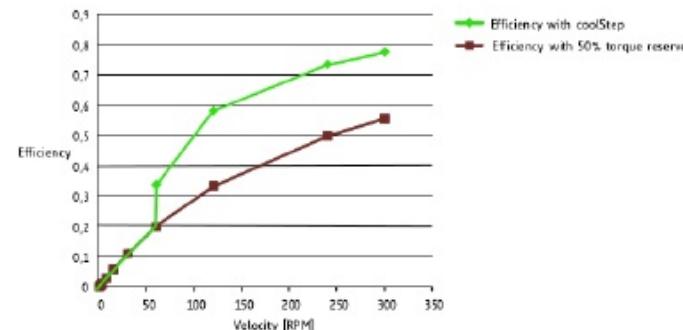


FIGURA 2.5. Función coolstep.⁵

Por último, se utilizó la función *dcStep*, que es un modo de commutación automático que ajusta la velocidad del motor en caso de existir sobrecarga en el eje, es decir que si no puede mover la carga acoplada al eje con la velocidad establecida, se ajusta a una velocidad menor para poder seguir en movimiento y no detenerse por completo.

El driver TMC5130 cuenta con cincuenta registros que se utilizan para configurar las funcionalidades del CI y controlar el motor. En el capítulo 3 se darán más detalles de los registros configurados con la ayuda del software TMCL-IDE.

2.3. Interfaz de usuario

Respecto a la interacción entre el usuario y el equipo surgió en reuniones con el cliente la necesidad de contar con una interfaz moderna, que permita a un usuario dentro de un laboratorio configurar al equipo a pie de máquina. Dando lugar al siguiente requerimiento:

- La configuración de la máquina debe poder realizarse a través de una pantalla táctil.

Se decidió trabajar con pantallas del tipo *HMI (Human Machine Interface)*. Las mismas se encargan exclusivamente del procesamiento gráfico. En general, cuentan con un software de diseño para la creación de la interfaz gráfica, es decir que permiten crear botones, barras, pantallas y diferentes tipos de objetos para interactuar con el usuario. Luego se le da funcionalidad a cada uno de estos objetos creados en el software y a través de un protocolo de comunicación se interactúa con el sistema de control, permitiendo finalmente controlar y configurar el equipo. En el caso de este equipo la pantalla se comunica con un microcontrolador,

⁵Imagen tomada de [9].

2.3. Interfaz de usuario

11

El efecto final es reducir la energía suministrada según hojas de datos [16] hasta un 75 %. Esto aplica incluso en equipos donde la carga es constante, como es el caso del dip coater, ya que la carga variable representada por un wafer de silicio o un portaobjeto es completamente despreciable.

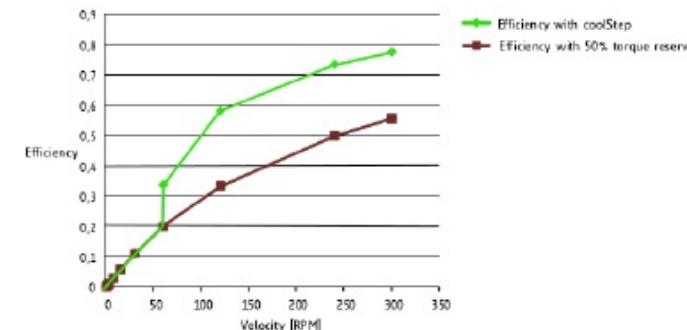


FIGURA 2.5. Función coolstep.⁵

Por último, se utilizó la función *dcStep*, que es un modo de commutación automático que ajusta la velocidad del motor en caso de existir sobrecarga en el eje, es decir que si no puede mover la carga acoplada al eje con la velocidad establecida, se ajusta a una velocidad menor para poder seguir en movimiento y no detenerse por completo.

El driver TMC5130 cuenta con cincuenta registros que se utilizan para configurar las funcionalidades del CI y controlar el motor. En el capítulo 3 se darán más detalles de los registros configurados con la ayuda del software TMCL-IDE.

2.3. Interfaz de usuario

Respecto a la interacción entre el usuario y el equipo surgió en reuniones con el cliente la necesidad de contar con una interfaz moderna, que permita a un usuario dentro de un laboratorio configurar al equipo a pie de máquina. Dando lugar al siguiente requerimiento:

- La configuración de la máquina debe poder realizarse a través de una pantalla táctil.

Se decidió trabajar con pantallas del tipo *HMI (Human Machine Interface)*. Las mismas se encargan exclusivamente del procesamiento gráfico. En general, cuentan con un software de diseño para la creación de la interfaz gráfica, es decir que permiten crear botones, barras, pantallas y diferentes tipos de objetos para interactuar con el usuario. Luego se le da funcionalidad a cada uno de estos objetos creados en el software y a través de un protocolo de comunicación se interactúa con el sistema de control, permitiendo finalmente controlar y configurar el equipo. En el caso de este equipo la pantalla se comunica con un microcontrolador,

⁵Imagen tomada de [11].

pero podría comunicarse con algún otro sistema de control como por ejemplo un PLC (*Programmable Logic Controller*).

Luego de una investigación de mercado se eligió a la empresa STONE [16]. El fabricante ofrece un catálogo amplio de pantallas que caracteriza según el tipo de aplicación y entorno de trabajo. Ofrece entonces pantallas para usos industriales, civiles o avanzados. Por las dimensiones finales del equipo y el tipo de uso se optó por pantallas avanzadas de 4.3 pulgadas. Se detallan en la tabla 2.1 las características técnicas de dos pantallas de 4.3 pulgadas del fabricante.

TABLA 2.1. Comparación pantallas táctiles Stone 4.3.

	STWI043WT	STVI043WT
CPU	Cortex A8	CortexM4
Refresh Rate	1G Hz	200 MHz
Image format	png, bmp, jpg, svg, gif	bmp, jpg
Resolution	480×272 pixel	480×272 pixel
Flash	256 MB	128 MB
Color	262 K	65 K
PCB	2.0 mm black, ROHS	1.6 mm green
Touch Type	Resistive	Resistive
Interface	RS232/RS422/RS485/TTL	RS232/RS485/TTL

El modelo elegido fue el STWI043WT, que pertenece a la nueva línea productos, tiene mayor capacidad de procesamiento, cuenta con un software nuevo de configuración con mayores funcionalidades respecto al utilizado por el otro modelo y la diferencia de precios no supera el 15 %.

La comunicación de la pantalla STWI043WT con el microcontrolador se estableció a través del protocolo serie.

2.4. Estructura mecánica

Se presentan a continuación los siguientes requerimientos asociados a las partes mecánicas del equipo:

- La estructura principal del equipo debe ser fabricada con perfil de aluminio anodizado natural.
- El recorrido mecánico de desplazamiento de muestra debe ser como mínimo de [3500 mm].
- Las piezas especiales del equipo deben ser mecanizadas en aluminio.

Se decidió trabajar con el proveedor Perfiles de Aluminio .NET [17], que cuenta con diferentes modelos y dimensiones de perfiles necesarios para la fabricación de la estructura.

El equipo cuenta con una guía lineal acoplada al perfil principal. Para la elección de la misma se tuvieron en cuenta las siguientes consideraciones:

1. El ambiente cambia según las soluciones químicas utilizadas. Es posible entonces que se trabaje con soluciones corrosivas que afecten la estructura.
2. El uso de lubricantes en las guías podría afectar la calidad del experimento.

pero podría comunicarse con algún otro sistema de control como por ejemplo un PLC (*Programmable Logic Controller*).

Luego de una investigación de mercado se eligió a la empresa STONE [18]. El fabricante ofrece un catálogo amplio de pantallas que caracteriza según el tipo de aplicación y entorno de trabajo. Ofrece entonces pantallas para usos industriales, civiles o avanzados. Por las dimensiones finales del equipo y el tipo de uso se optó por pantallas avanzadas de 4.3 pulgadas. Se detallan en la tabla 2.1 las características técnicas de dos pantallas de 4.3 pulgadas del fabricante.

TABLA 2.1. Comparación pantallas táctiles Stone 4.3.

	STWI043WT	STVI043WT
CPU	Cortex A8	CortexM4
Refresh Rate	1G Hz	200 MHz
Image format	png, bmp, jpg, svg, gif	bmp, jpg
Resolution	480×272 pixel	480×272 pixel
Flash	256 MB	128 MB
Color	262 K	65 K
PCB	2.0 mm black, ROHS	1.6 mm green
Touch Type	Resistive	Resistive
Interface	RS232/RS422/RS485/TTL	RS232/RS485/TTL

El modelo elegido fue el STWI043WT, que pertenece a la nueva línea productos, tiene mayor capacidad de procesamiento, cuenta con un software nuevo de configuración con mayores funcionalidades respecto al utilizado por el otro modelo y la diferencia de precios no supera el 15 %.

La comunicación de la pantalla STWI043WT con el microcontrolador se estableció a través del protocolo serie.

2.4. Estructura mecánica

Se presentan a continuación los siguientes requerimientos asociados a las partes mecánicas del equipo:

- La estructura principal del equipo debe ser fabricada con perfil de aluminio anodizado natural.
- El recorrido mecánico de desplazamiento de muestra debe ser como mínimo de [3500 mm].
- Las piezas especiales del equipo deben ser mecanizadas en aluminio.

Se decidió trabajar con el proveedor Perfiles de Aluminio .NET [19], que cuenta con diferentes modelos y dimensiones de perfiles necesarios para la fabricación de la estructura.

El equipo cuenta con una guía lineal acoplada al perfil principal. Para la elección de la misma se tuvieron en cuenta las siguientes consideraciones:

1. El ambiente cambia según las soluciones químicas utilizadas. Es posible entonces que se trabaje con soluciones corrosivas que afecten la estructura.
2. El uso de lubricantes en las guías podría afectar la calidad del experimento.

2.5. Sistema electrónico propuesto

13

3. Se deben evitar vibraciones en la estructura para no dañar la calidad del film.

Se decidió entonces trabajar con la empresa IGUS [18], que se especializa en la fabricación de polímeros. La misma ofrece guías lineales que se deslizan en lugar de rodar, y por lo tanto no utilizan rodamientos metálicos. Los polímeros están combinados con materiales anticorrosivos y no requieren de la aplicación de lubricante, es decir, que conforman un entorno de trabajo limpio y libre de mantenimiento periódico. Se observa en la figura 2.6 cuatro tipos de guías en donde se puede apreciar el polímero auto-lubricado que se ubica entre el eje y el carro.



FIGURA 2.6. Guía Lineal IGUS.⁶

Para el diseño y fabricación de piezas mecanizadas en aluminio se trabajó con el software BOBCAD [19] CAD/CAM (Computer-Aided Design /Computer-Aided Manufacturing). Un software utilizado en la industria manufacturera, que se encuentra constituido por dos módulos fundamentales que permiten abarcar aspectos de diseño y modelado de pieza y luego de fabricación.

Con la parte CAD se diseña el modelo 3D de la pieza. Con el fin de corregir errores de diseño con mayor velocidad, se realiza una impresión 3D con filamento plástico para probar las dimensiones y la factibilidad técnica de la pieza. Una vez que el modelo en su versión plástica queda aprobado, se comienza con la configuración del módulo CAM, este módulo se encarga de convertir, a través de diferentes estrategias, al modelo 3D en lenguaje de máquina que el equipo puede interpretar. Se observa en la imagen 2.7 la fresadora CNC de la marca FAGOR [20] utilizada para la fabricación de las piezas del equipo dip coater. El control de la fresadora interpreta el código G-CODE también conocido como RS-274 [21] generado por el modulo CAM y lo convierte en movimientos de motores.

2.5. Sistema electrónico propuesto

Se presentan a continuación los siguientes requerimientos:

- El sistema debe permitir que el usuario pueda configurar en un programa variables de desplazamiento y tiempos de espera.
- Un programa previamente configurado debe poder ejecutarse o guardarse en memoria interna.

⁶Imagen tomada de [18].

⁷Imagen tomada en el centro tecnológico de FUNINTEC.

2.5. Sistema electrónico propuesto

13

3. Se deben evitar vibraciones en la estructura para no dañar la calidad del film.

Se decidió entonces trabajar con la empresa IGUS [20], que se especializa en la fabricación de polímeros. La misma ofrece guías lineales que se deslizan en lugar de rodar, y por lo tanto no utilizan rodamientos metálicos. Los polímeros están combinados con materiales anticorrosivos y no requieren de la aplicación de lubricante, es decir, que conforman un entorno de trabajo limpio y libre de mantenimiento periódico. Se observa en la figura 2.6 cuatro tipos de guías en donde se puede apreciar el polímero auto-lubricado que se ubica entre el eje y el carro.



FIGURA 2.6. Guía Lineal IGUS.⁶

Para el diseño y fabricación de piezas mecanizadas en aluminio se trabajó con el software BOBCAD [21] CAD/CAM (Computer-Aided Design /Computer-Aided Manufacturing). Un software utilizado en la industria manufacturera, que se encuentra constituido por dos módulos fundamentales que permiten abarcar aspectos de diseño y modelado de pieza y luego de fabricación.

Con la parte CAD se diseña el modelo 3D de la pieza. Con el fin de corregir errores de diseño con mayor velocidad, se realiza una impresión 3D con filamento plástico para probar las dimensiones y la factibilidad técnica de la pieza. Una vez que el modelo en su versión plástica queda aprobado, se comienza con la configuración del módulo CAM, este módulo se encarga de convertir, a través de diferentes estrategias, al modelo 3D en lenguaje de máquina que el equipo puede interpretar. Se observa en la imagen 2.7 la fresadora CNC de la marca FAGOR [22] utilizada para la fabricación de las piezas del equipo dip coater. El control de la fresadora interpreta el código G-CODE también conocido como RS-274 [23] generado por el modulo CAM y lo convierte en movimientos de motores.

2.5. Sistema electrónico propuesto

Se presentan a continuación los siguientes requerimientos:

- El sistema debe permitir que el usuario pueda configurar en un programa variables de desplazamiento y tiempos de espera.
- Un programa previamente configurado debe poder ejecutarse o guardarse en memoria interna.

⁶Imagen tomada de [20].

⁷Imagen tomada en el centro tecnológico de FUNINTEC.

FIGURA 2.7. Fresadora Fagor GVC 600.⁷

- El usuario debe poder guardar al menos 10 programas en la memoria no volátil del sistema.
- Se debe utilizar control de versionado de cambios durante el desarrollo del firmware.
- El desarrollo del firmware debe realizarse con capas de abstracción de software de tal manera que permita en un futuro cambiar de microcontrolador sin mayor esfuerzo.
- El desarrollo se realizará sobre un módulo microcontrolador ESP32.
- Se deben registrar variables de humedad, presión y temperatura [opcional].

De reuniones con el cliente y de la necesidad de trabajar con un equipo que permita a futuro contar con una comunicación Wi-Fi, surgió el requerimiento de trabajar con el módulo ESP32 [22]. El ESP32 es un módulo del tipo *SoC (System On Chip)*, es decir que además del microcontrolador y sus periféricos internos, agrega periféricos externos para brindar conectividad inalámbrica y almacenamiento extra para datos y programa.

Teniendo en cuenta los requerimientos analizados en las secciones previas y los requerimientos analizados en esta sección se propuso, como se observa en la figura 2.8, el siguiente esquema de equipo dip coater.

El equipo estará entonces compuesto por el módulo ESP32-WROOM como unidad central de procesamiento. Contará con dos canales de comunicación serie, uno para establecer una consola de comandos que permita comunicar al equipo con una computadora, realizar pruebas de funcionamiento y configuraciones, y el otro para comunicarse con la pantalla táctil. También establecerá un protocolo de almacenamiento para guardar los programas que el usuario cree en la memoria FLASH disponible. Finalmente contará con un módulo de comunicación sobre protocolo I2C para obtener datos del sensor de humedad y temperatura BME280.

FIGURA 2.7. Fresadora Fagor GVC 600.⁷

- El usuario debe poder guardar al menos 10 programas en la memoria no volátil del sistema.
- Se debe utilizar control de versionado de cambios durante el desarrollo del firmware.
- El desarrollo del firmware debe realizarse con capas de abstracción de software de tal manera que permita en un futuro cambiar de microcontrolador sin mayor esfuerzo.
- El desarrollo se realizará sobre un módulo microcontrolador ESP32.
- Se deben registrar variables de humedad, presión y temperatura [opcional].

De reuniones con el cliente y de la necesidad de trabajar con un equipo que permita a futuro contar con una comunicación Wi-Fi, surgió el requerimiento de trabajar con el módulo ESP32 [24]. El ESP32 es un módulo del tipo *SoC (System On Chip)*, es decir que además del microcontrolador y sus periféricos internos, agrega periféricos externos para brindar conectividad inalámbrica y almacenamiento extra para datos y programa.

Teniendo en cuenta los requerimientos analizados en las secciones previas y los requerimientos analizados en esta sección se propuso, como se observa en la figura 2.8, el siguiente esquema de equipo dip coater.

El equipo estará entonces compuesto por el módulo ESP32-WROOM como unidad central de procesamiento, contará con dos comunicaciones a través del periférico UART, una para establecer una consola de comandos que permita comunicar al equipo con una computadora y realizar pruebas de funcionamiento y configuraciones y la otra para comunicarse con la pantalla táctil. También establecerá un protocolo de almacenamiento para guardar los programas que el usuario cree en la memoria FLASH disponible. Finalmente contará con un módulo de comunicación sobre protocolo I2C para obtener datos del sensor de humedad y temperatura BME280.

2.6. Herramientas de desarrollo

15

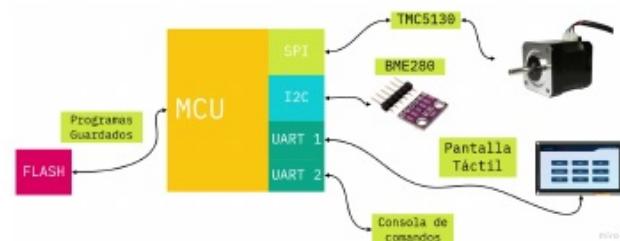


FIGURA 2.8. Esquema de equipo propuesto.

2.6. Herramientas de desarrollo

Para la implementación del hardware se utilizó el software libre de diseño de circuitos impresos KICAD [10], la elección del mismo se basó en los siguientes puntos:

- Las capacidades que brinda el software son suficientes para el desarrollo de este hardware.
- Se valora el apoyo del CERN:European Organization for Nuclear Research [23] al proyecto KICAD.

Para la implementación del firmware se trabajó con el *framework* ESP-IDF [24] provisto por el fabricante del microcontrolador. Dicho entorno se ejecuta sobre FreeRTOS, que es un sistema operativo de tiempo real utilizado en dispositivos embebidos que permite un desarrollo de software bajo un esquema multi-tareas.

Se trabajó con el entorno de programación ECLIPSE IDE, la elección se basó en los siguientes puntos:

- El fabricante del microcontrolador ESPRESSIF ofrece *pluggings* para incorporar al entorno y facilitar el desarrollo.
- Existe documentación para la configuración del *framework* ESP-IDF [25] sobre el entorno.

2.6. Herramientas de desarrollo

15

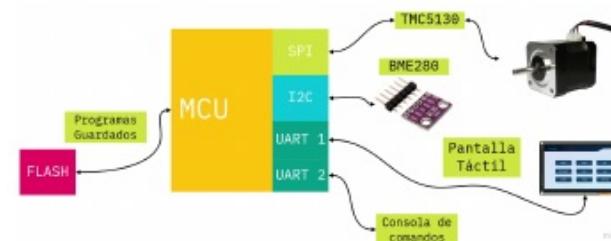


FIGURA 2.8. Esquema de equipo propuesto.

2.6. Herramientas de desarrollo

Para la implementación del hardware se utilizó el software libre de diseño de circuitos impresos KICAD [12], la elección del mismo se basó en los siguientes puntos:

- Las capacidades que brinda el software son suficientes para el desarrollo de este hardware.
- Se valora el apoyo del CERN:European Organization for Nuclear Research [25] al proyecto KICAD.
- Las últimas versiones presentan mejoras significativas respecto a sus predecesoras.

Para la implementación del firmware se trabajó con el *framework* ESP-IDF [31] provisto por el fabricante del microcontrolador. Dicho entorno se ejecuta sobre FreeRTOS, que es un sistema operativo de tiempo real utilizado en dispositivos embebidos que permite un desarrollo de software bajo un esquema multi-tareas.

Se trabajó con el entorno de programación ECLIPSE IDE, la elección se basó en los siguientes puntos:

- El fabricante del microcontrolador ESPRESSIF ofrece *pluggings* para incorporar al entorno y facilitar el desarrollo.
- Existe documentación para la configuración del *framework* ESP-IDF [26] sobre el entorno.

Capítulo 3

Diseño e Implementación

En el siguiente capítulo se presenta el diseño y la implementación de las tres partes fundamentales del equipo. Se abordan aspectos de diseño de hardware, desarrollo de firmware y diseño y fabricación mecánica.

3.1. Hardware

3.1.1. Diseño basado en módulos de hardware libre

El diseño de la placa electrónica se basó en el estudio de los siguientes módulos:

- TMC5130-EVAL [15]
- NodeMCU [26]

Se destaca que ambos proyectos adhieren a la filosofía del hardware libre. Por lo tanto se pudieron descargar y estudiar los diagramas esquemáticos de ambas placas. Las primeras pruebas de implementación de hardware se realizaron interconectando ambos módulos como puede observarse en la figura 3.1.

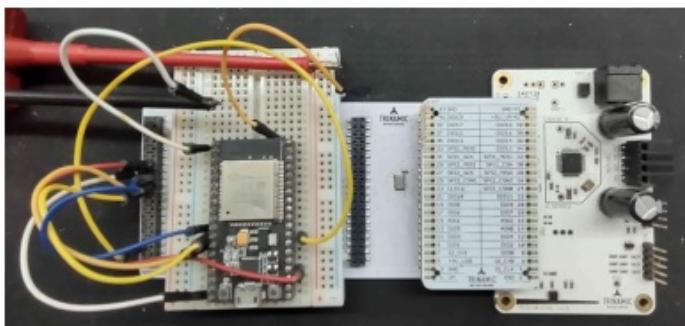


FIGURA 3.1. Módulo NodeMCU + TMC5130-EVAL.

3.1.2. Etapa de alimentación

Se observa en la figura 3.2 la etapa reguladora de tensión que permite alimentar al equipo con tensiones continuas de entre 24 V y 46 V. En el diseño se utilizó el CI LM5161 en modo *step-down buck converter*, una configuración que permite bajar la

Capítulo 3

Diseño e Implementación

En el siguiente capítulo se presenta el diseño y la implementación de las tres partes fundamentales del equipo. Se abordan aspectos de diseño de hardware, desarrollo de firmware y diseño y fabricación mecánica.

3.1. Hardware

3.1.1. Diseño basado en módulos de hardware libre

El diseño de la placa electrónica se basó en el estudio de los siguientes módulos:

- TMC5130-EVAL [17]
- NodeMCU [27]

Se destaca que ambos proyectos adhieren a la filosofía del hardware libre. Por lo tanto se pudieron descargar y estudiar los diagramas esquemáticos de ambas placas. Las primeras pruebas de implementación de hardware se realizaron interconectando ambos módulos como puede observarse en la figura 3.1.

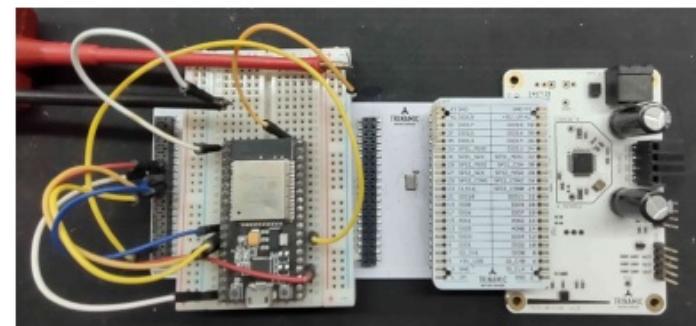


FIGURA 3.1. Módulo NodeMCU + TMC5130-EVAL.

3.1.2. Etapa de alimentación

Se observa en la figura 3.2 la etapa reguladora de tensión que permite alimentar al equipo con tensiones continuas de entre 24 V y 46 V. En el diseño se utilizó el CI LM5161 en modo *step-down buck converter*, una configuración que permite bajar la

tensión de entrada según la relación de las resistencias de feedback. En este caso se configuró una salida de 5 V la cual tiene una eficiencia energética de 86 % según su hoja de datos. Luego a través de un regulador de tensión se obtienen 3,3 V que se utilizan para alimentar el microcontrolador.

Etapas Reguladoras de Tensión
Entrada 24–48 Volt
Salida 5V hacia Integrado TMC130
Salida 3.3V hacia microcontrolador con diodo de protección

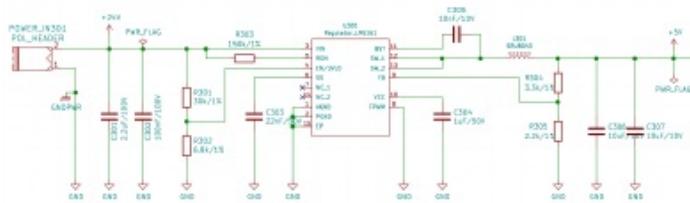


FIGURA 3.2. Módulo de entrada.

El equipo se diseñó para ser alimentado con una fuente externa de tensión continua, simplificando así cuestiones regulatorias de certificación que deben cumplir equipos que se alimentan directamente a la red eléctrica.

3.1.3. Etapa de comunicación

El módulo NodeMCU es una placa de desarrollo que contiene el SoC ESP32-WROOM. A partir del estudio de su diseño, se implementó la etapa de conversión SERIAL-USB que puede observarse en la figura 3.3.

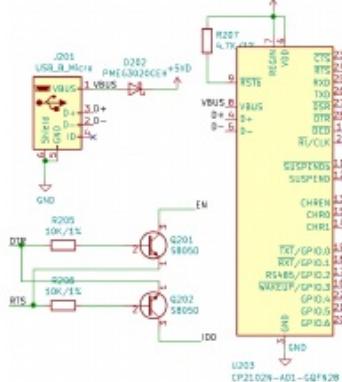


FIGURA 3.3. Conversor serie-USB.

Mantener esta etapa de entrada en la placa electrónica final habilita la conexión directa del equipo dip coater con un puerto USB de computadora, lo cual permite

tensión de entrada según la relación de las resistencias de feedback. En este caso se configuró una salida de 5 V la cual tiene una eficiencia energética de 86 % según su hoja de datos. Luego a través de un regulador de tensión se obtienen 3,3 V que se utilizan para alimentar el microcontrolador.

Etapas Reguladoras de Tensión
Entrada 24–48 Volt
Salida 5V hacia Integrado TMC130
Salida 3.3V hacia microcontrolador con diodo de protección

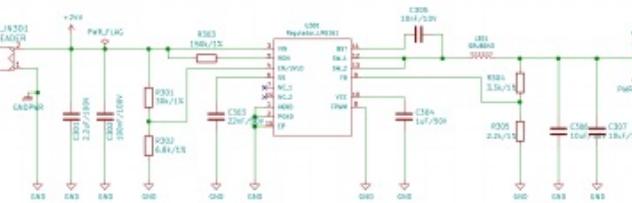


FIGURA 3.2. Módulo de entrada.

El equipo se diseñó para ser alimentado con una fuente externa de tensión continua, simplificando así cuestiones regulatorias de certificación que deben cumplir equipos que se alimentan directamente a la red eléctrica.

3.1.3. Etapa de comunicación

El módulo NodeMCU es una placa de desarrollo que contiene el SoC ESP32-WROOM. A partir del estudio de su diseño, se implementó la etapa de conversión SERIAL-USB que puede observarse en la figura 3.3.

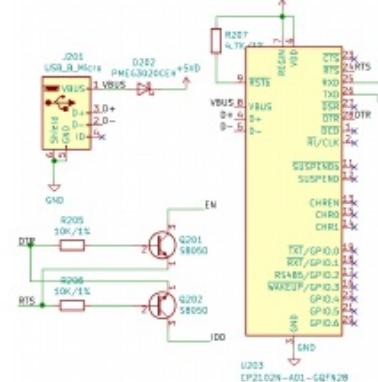


FIGURA 3.3. Conversor serie-USB.

Mantener esta etapa de entrada en la placa electrónica final habilita la conexión directa del equipo dip coater con un puerto USB de computadora, lo cual permite

establecer una comunicación constante con el periférico UART y descargar un firmware nuevo sin tener que contar con un programador externo. Se utilizó para esta etapa de conversión el CI CP2102.

Para descargar un firmware en el microcontrolador ESP32 se debe seguir la siguiente secuencia:

1. Mantener la terminal IO0 en GND.
2. Poner la terminal EN en GND para reiniciar el mismo.
3. El microcontrolador se inicia en modo *boot*.
4. Enviar firmware desde el entorno de desarrollo.

La interfaz UART CP2102 consta de señales de transmisión y recepción de datos TX y RX. También admite las señales de control RTS/CTS, DSR/DTR y X-ON/X-OFF. Se incorporó en el diseño el uso de las terminales DTR y RTS para generar la secuencia de descarga de manera automática. Sin embargo la misma presentó cierta inestabilidad y no siempre se logró generar la secuencia correctamente, por tal motivo se implementará en la nueva versión de la placa botones para poder forzar dicha secuencia.

3.1.4. Driver TMC5130

El módulo TMC5130-EVAL, como se describió en la sección 2.2, contiene al CI TMC5130. Del estudio de esta placa de evaluación se extrajeron las configuraciones necesarias para lograr la correcta utilización del driver. Se tuvieron en cuenta las recomendaciones de diseño establecidas por el fabricante, como por ejemplo la incorporación de un clock externo de 16 MHz, que se puede observar en la figura 3.4 el cual es necesario en aplicaciones de alta precisión.

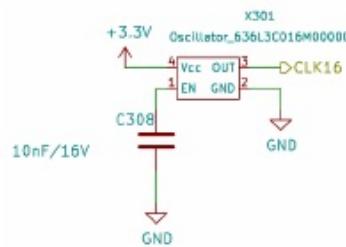


FIGURA 3.4. Clock para el CI TMC5130.

El driver se alimenta por las terminales VS/VSA con la tensión de entrada de la placa electrónica y por la terminal VCC con 5 V, los cuales son generados internamente por un regulador en la terminal 5VOUT. Si bien esta configuración funciona, se pretende mejorar el diseño en la próxima versión de la placa. El fabricante sugiere alimentar la terminal VCC con una tensión externa para evitar el uso del regulador de tensión interno y mejorar así la eficiencia térmica del CI.

Se observa en la figura 3.5 las conexiones del driver con el motor paso a paso, la etapa de alimentación y las conexiones del puerto SPI para la comunicación con el microcontrolador.

establecer una comunicación constante con el periférico UART y descargar un firmware nuevo sin tener que contar con un programador externo. Se utilizó para esta etapa de conversión el CI CP2102.

Para descargar un firmware en el microcontrolador ESP32 se debe seguir la siguiente secuencia:

1. Mantener la terminal IO0 en GND.
2. Poner la terminal EN en GND para reiniciar el mismo.
3. El microcontrolador se inicia en modo *boot*.
4. Enviar firmware desde el entorno de desarrollo.

La interfaz UART CP2102 consta de señales de transmisión y recepción de datos TX y RX respectivamente, también admite las señales de control RTS/CTS, DS-R/DTR y X-ON/X-OFF. Se incorporó en el diseño el uso de las terminales DTR y RTS para generar la secuencia de descarga de manera automática. Sin embargo la misma presentó cierta inestabilidad y no siempre se logró generar la secuencia correctamente, por tal motivo se implementará en la nueva versión de la placa botones para poder forzar dicha secuencia.

3.1.4. Driver TMC5130

El módulo TMC5130-EVAL, como se describió en la sección 2.2, contiene al CI TMC5130. Del estudio de esta placa de evaluación se extrajeron las configuraciones necesarias para lograr la correcta utilización del driver. Se tuvieron en cuenta las recomendaciones de diseño establecidas por el fabricante, como por ejemplo la incorporación de un clock externo de 16 MHz, que se puede observar en la figura 3.4 el cual es necesario en aplicaciones de alta precisión.

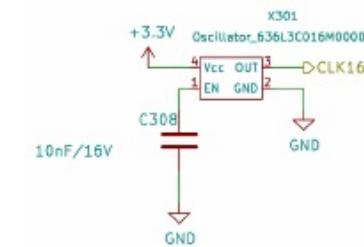


FIGURA 3.4. Clock para el CI TMC5130.

El driver se alimenta por las terminales VS/VSA con la tensión de entrada de la placa electrónica y por la terminal VCC con 5 V, los cuales son generados internamente por un regulador en la terminal 5VOUT. Si bien esta configuración funciona, se pretende mejorar el diseño en la próxima versión de la placa. El fabricante sugiere alimentar la terminal VCC con una tensión externa para evitar el uso del regulador de tensión interno y mejorar así la eficiencia térmica del CI.

Se observa en la figura 3.5 las conexiones del driver con el motor paso a paso, la etapa de alimentación y las conexiones del puerto SPI para la comunicación con el microcontrolador.

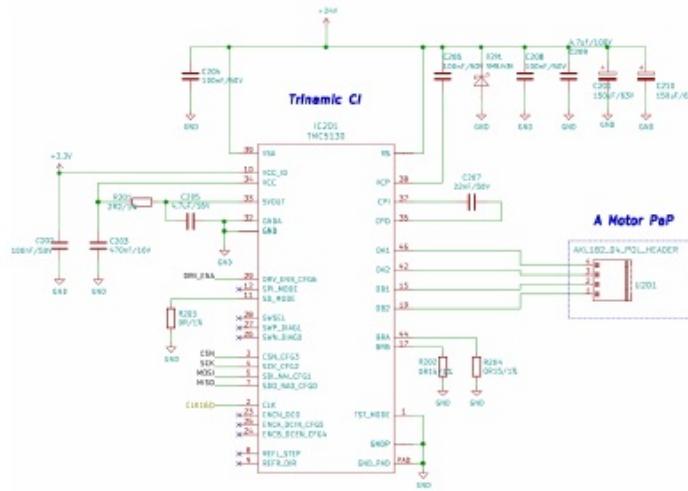


FIGURA 3.5. CI TMC5130

3.1.5. Diseño final

Finalmente, se observa en la figura 3.6 el diseño 3D generado por el software KICAD.

Todo el diseño y material asociado se encuentra disponible en el repositorio de la empresa TECSCI [4]. La placa electrónica de este equipo dip coater cuenta con una licencia CERN OHL v.1.2 [27].



FIGURA 3.6. Modelo 3D Kicad.

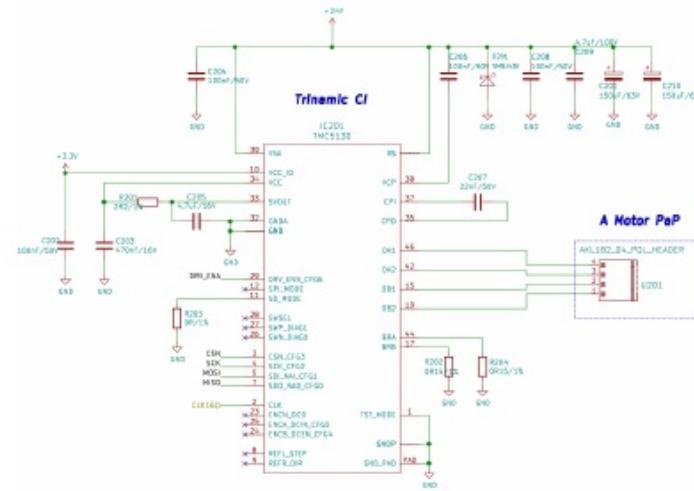


FIGURA 3.5. CI TMC5130

3.1.5. Diseño final

Finalmente, se observa en la figura 3.6 el diseño 3D generado por el software KICAD.

Todo el diseño y material asociado se encuentra disponible en el repositorio de la empresa TECSCI [4]. La placa electrónica de este equipo dip coater cuenta con una licencia CERN OHL v.1.2 [28].



FIGURA 3.6. Modelo 3D Kicad

3.1.6. Fabricación

La placa electrónica se fabricó con el proveedor local de circuitos impresos Ernesto Mayer S.A. [28]. A continuación se presenta la información de diseño de la misma y se describen restricciones impuestas por el propio fabricante:

- Grilla de posicionamiento principal: 0,25 mm.
- Grilla de ruteo principal: 0,25 mm.
- Agujeros de montaje: 3,2 mm.
- Pistas principales: 0,5 mm.
- Pistas inferiores: 0,25 mm, con límite particular 0,20mm (8 mils).
- Pistas superiores: 0,8 mm.
- Vías: 0,8 mm /0,4 mm, con límite particular 0,20mm (8 mils).
- Margen general: 0,22 mm.
- Margen particular: 0,2 mm, con límite particular 0,20 mm (8 mils).
- Fabricación: espesor 1,6mm FR4.
- Restricciones generales del fabricante: con límite estándar 0,254 mm (10 mils).

Luego de fabricar el PCB, se continuó con el montaje de componentes electrónicos superficiales, que estuvo a cargo de la empresa Asembli S.A. [29]. Se fabricó un primer lote de cinco placas. En la figura 3.7 se observa la placa con los componentes montados.



FIGURA 3.7. Plaqueta electrónica final.

3.1.6. Fabricación

La placa electrónica se fabricó con el proveedor local de circuitos impresos Ernesto Mayer S.A. [29]. A continuación se presenta la información de diseño de la misma y se describen restricciones impuestas por el propio fabricante:

- Grilla de posicionamiento principal: 0,25 mm.
- Grilla de ruteo principal: 0,25 mm.
- Agujeros de montaje: 3,2 mm.
- Pistas principales: 0,5 mm.
- Pistas inferiores: 0,25 mm, con límite particular 0,20mm (8 mils).
- Pistas superiores: 0,8 mm.
- Vías: 0,8 mm /0,4 mm, con límite particular 0,20mm (8 mils).
- Margen general: 0,22 mm.
- Margen particular: 0,2 mm, con límite particular 0,20 mm (8 mils).
- Fabricación: espesor 1,6mm FR4.
- Restricciones generales del fabricante: con límite estándar 0,254 mm (10 mils).

Luego de fabricar el PCB, se continuó con el montaje de componentes electrónicos superficiales, que estuvo a cargo de la empresa Asembli S.A. [30]. Se fabricó un primer lote de cinco placas. En la figura 3.7 se observa la placa con los componentes montados.

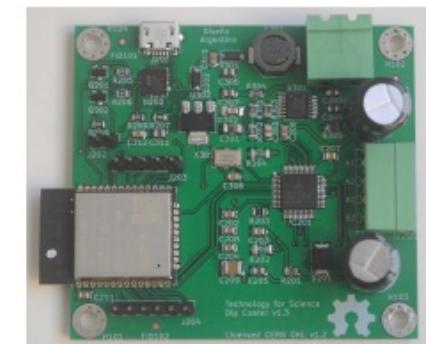


FIGURA 3.7. Plaqueta electrónica final.

3.2. Firmware

3.2.1. Capas de abstracción

Se desarrolló un firmware modular que permite incorporar código de manera incremental y ordenada. Se observan en la figura 3.8 las capas de abstracción de software implementadas. Esta **solución** tiene dos ideas fundamentales:

- No permitir llamados a funciones entre capas discontinuas. Los módulos de la capa superior o capa APP solo pueden hacer llamados a funciones de la capa intermedia o capa API (*Application Programming Interfaces*) y estas últimas solo pueden llamar a funciones de la capa inferior o capa BOARD.
- Las aplicaciones funcionan de manera independiente, las mismas se pueden habilitar o deshabilitar individualmente.

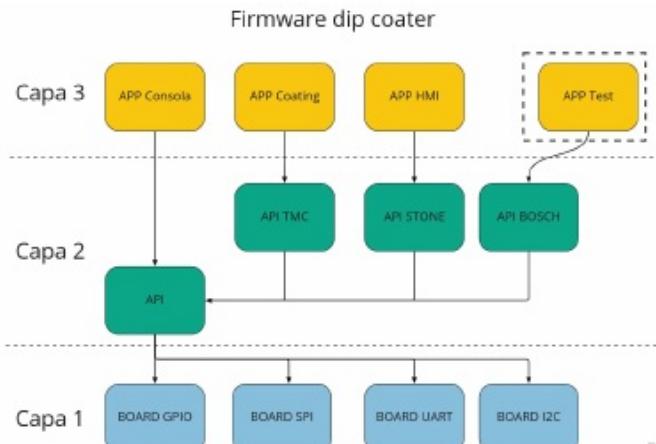


FIGURA 3.8. Capas de abstracción de software.

La capa tres corresponde a la capa de aplicaciones. El firmware cuenta con 3 aplicaciones fundamentales para el funcionamiento del equipo y una aplicación de testing utilizada para probar nuevos componentes de software. Cada aplicación contiene al menos una **tarea** del sistema operativo freeRTOS. A continuación se detallan las aplicaciones:

- APP coating: se encarga de la comunicación con el driver TMC5130, de la ejecución de movimientos individuales y del proceso completo de dip coating.
- APP consola: administra una consola de comandos que permite ejecutar y configurar el equipo a través de una comunicación **serial**. **Recibe** y procesa comandos de la consola y los envía a través de una **cola** de freeRTOS a la app coating.

3.2. Firmware

3.2.1. Capas de abstracción

Se desarrolló un firmware modular que permite incorporar código de manera incremental y ordenada. Se observan en la figura 3.8 las capas de abstracción de software implementadas. Esta **implementación** tiene dos ideas fundamentales:

- No permitir llamados a funciones entre capas discontinuas. Los módulos de la capa superior o capa APP solo pueden hacer llamados a funciones de la capa intermedia o capa API (*Application Programming Interfaces*) y estas últimas solo pueden llamar a funciones de la capa inferior o capa BOARD.
- Las aplicaciones funcionan de manera independiente, las mismas se pueden habilitar o deshabilitar individualmente.

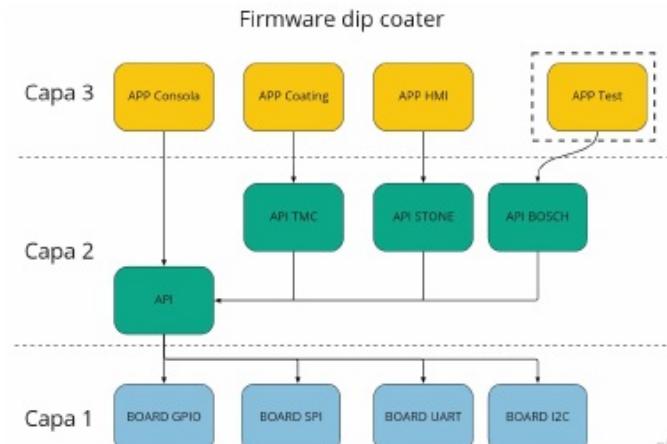


FIGURA 3.8. Capas de abstracción de software.

La capa tres corresponde a la capa de aplicaciones. El firmware cuenta con 3 aplicaciones fundamentales para el funcionamiento del equipo y una aplicación de testing utilizada para probar nuevos componentes de software. Cada aplicación contiene al menos una **task** del sistema operativo freeRTOS. A continuación se detallan las aplicaciones:

- APP coating: se encarga de la comunicación con el driver TMC5130, de la ejecución de movimientos individuales y del proceso completo de dip coating.
- APP consola: administra una consola de comandos que permite ejecutar y configurar el equipo a través de una comunicación **serial**, **recibe** y procesa comandos de la consola y los envía a través de una **queue** de freeRTOS a la app coating.

- APP hmi: administra la interfaz de configuración, establece una comunicación serial con la pantalla táctil, recibe comandos a través de una *cola* de freeRTOS y los envía a la app coating para ser procesados.
- APP test: se utiliza para probar nuevos componentes y realizar *test* sobre el sistema.

La capa dos esta compuesta por bloques de código provistos por los fabricantes de drivers:

- API TMC: provista por el fabricante Trinamic y adaptada para ser ejecutada bajo el framework ESP-IDF.
- API BOSH: provista por el fabricante y adaptada a este firmware.
- API STONE: contiene los módulos de software que interaccionan con la pantalla táctil.
- API: se utiliza como puente hacia la capa BOARD que tiene acceso a los periféricos del microcontrolador y con módulos de software del *framework* ESP-IDF.

La capa uno es la que interacciona con los módulos de hardware del microcontrolador. Por lo tanto, esta capa es la única que contiene llamados a funciones disponibles en el framework ESP-IDF, como por ejemplo las funciones de configuración de los periféricos UART, GPIO, SPI, etc. Es importante mencionar que si debido a un rediseño se decide cambiar el microcontrolador utilizado, solo se deberán reescribir los módulos pertenecientes a esta capa y se podrá mantener el resto del programa sin alteraciones.

3.2.2. Módulos principales de software

Funcionamiento general

El equipo dip coater puede ser utilizado de dos maneras diferentes:

1. A través de una consola de comandos.
2. A través de una pantalla táctil.

Ambas opciones permiten ejecutar un ciclo completo de dip coating y también permiten ejecutar comandos de manera individual. Las aplicaciones app consola y app hmi se encargan de la interacción con el usuario y son las encargadas de enviar a la aplicación app coating los comandos para ejecutar los distintos movimientos.

Control de movimientos

La aplicación app coating contiene toda la lógica de control de movimientos. La misma se encarga de realizar la configuración inicial del driver TMC5130, de ejecutar los procesos completos de dip coating y de procesar comandos individuales para generar diferentes tipos de acciones.

Como se mencionó en la subsección 2.2.1 calcular los parámetros iniciales del driver es una tarea compleja, por lo que se optó por utilizar el software TMCL-IDE provisto por el fabricante y realizar la configuración de los registros de manera interactiva.

- APP hmi: administra la interfaz de configuración, establece una comunicación serial con la pantalla táctil, recibe comandos a través de una *queue* de freeRTOS y los envía a la app coating para ser procesados.
- APP test: se utiliza para probar nuevos componentes y realizar *test* sobre el sistema.

La capa dos esta compuesta por bloques de código provistos por los fabricantes de drivers:

- API TMC: provista por el fabricante Trinamic y adaptada para ser ejecutada bajo el framework ESP-IDF.
- API BOSH: provista por el fabricante y adaptada a este firmware.
- API STONE: contiene los módulos de software que interaccionan con la pantalla táctil.
- API: se utiliza como puente hacia la capa BOARD que tiene acceso a los periféricos del microcontrolador y con módulos de software del *framework* ESP-IDF.

La capa uno es la que interacciona con los módulos de hardware del microcontrolador. Por lo tanto, esta capa es la única que contiene llamados a funciones disponibles en el framework ESP-IDF, como por ejemplo las funciones de configuración de los periféricos UART, GPIO, SPI, etc. Es importante mencionar que si debido a un rediseño se decide cambiar el microcontrolador utilizado, solo se deberán reescribir los módulos pertenecientes a esta capa y se podrá mantener el resto del programa sin alteraciones.

3.2.2. Módulos principales de software

Funcionamiento general

El equipo dip coater puede ser utilizado de dos maneras diferentes:

1. A través de una consola de comandos.
2. A través de una pantalla táctil.

Ambas opciones permiten ejecutar un ciclo completo de dip coating y también permiten ejecutar comandos de manera individual. Las aplicaciones app consola y app hmi se encargan de la interacción con el usuario y son las encargadas de enviar a la aplicación app coating los comandos para ejecutar los distintos movimientos.

Control de movimientos

La aplicación app coating contiene toda la lógica de control de movimientos. La misma se encarga de realizar la configuración inicial del driver TMC5130, de ejecutar los procesos completos de dip coating y de procesar comandos individuales para generar diferentes tipos de acciones.

Como se mencionó en la subsección 2.2.1 calcular los parámetros iniciales del driver es una tarea compleja, por lo que se optó por utilizar el software TMCL-IDE provisto por el fabricante y realizar la configuración de los registros de manera interactiva.

En esta etapa de configuración es recomendable que el motor este acoplado al eje lineal, junto con el tornillo, la tuerca y el carro ya que el driver registra la corriente que circula por las bobinas del motor y calcula la fuerza contraelectromotriz que el eje esta ejerciendo. En la figura 3.9 se observa el entorno de desarrollo TMCL-IDE.

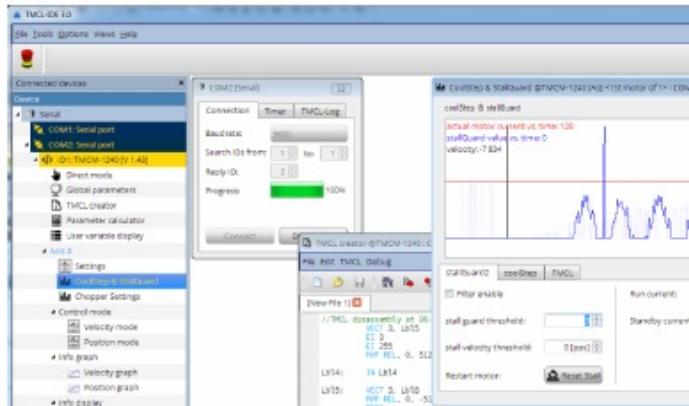


FIGURA 3.9. Software TMCL-IDE.

En la figura 3.10 se puede observar el *wizard* de configuración de las funciones *stallguard2* y *coolstep*.

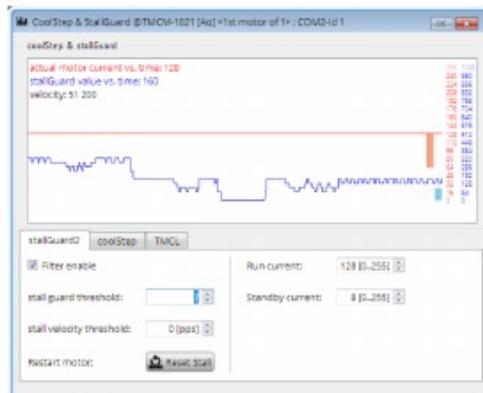


FIGURA 3.10. Configuración de funcionalidades stallguard2 y coolstep.

Como se mencionó en el capítulo 2 es posible usar *stallguard2* para detectar los límites mecánicos de recorrido. El parámetro *stall guard threshold* relaciona la **fuerza** contraelectromotriz registrada por el driver. A medida que se detecta mayor fuerza, es decir mayor oposición al movimiento y mayor corriente en los motores, el valor de *stallguard2* disminuye. Se debe configurar entonces un valor límite de detección para que una vez alcanzado se genere un evento.

En esta etapa de configuración es recomendable que el motor este acoplado al eje lineal, junto con el tornillo, la tuerca y el carro ya que el driver registra la corriente que circula por las bobinas del motor y calcula la fuerza contraelectromotriz que el eje esta ejerciendo. En la figura 3.9 se observa el entorno de desarrollo TMCL-IDE.

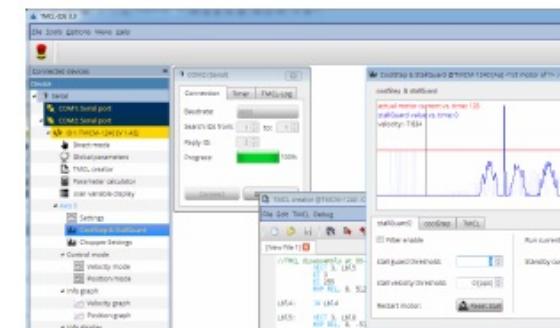


FIGURA 3.9. Software TMCL-IDE.

En la figura 3.10 se puede observar el *wizard* de configuración de las funciones *stallguard2* y *coolstep*. Como se mencionó en el capítulo 2 es posible usar *stallguard2* para detectar los límites mecánicos de recorrido. El parámetro *stall guard threshold* relaciona la **fuerza** contraelectromotriz registrada por el driver. A medida que se detecta mayor fuerza, es decir mayor oposición al movimiento y mayor corriente en los motores, el valor de *stallguard2* disminuye. Se debe configurar entonces un valor límite de detección para que una vez alcanzado se genere un evento.

El firmware cuenta con una función llamada *mod-coating-process-cero-machine* que utiliza *stallguard2*. Se encarga de detectar el evento que el driver envía cuando se detecta un final de carrera. La misma se encarga de parar el motor, generar un desplazamiento en sentido contrario y establecer una nueva posición. Esta función se ejecuta al encender el equipo y sirve para posicionar el carro del equipo.

Otros parámetros importantes a configurar son los que definen la rampa de aceleración del desplazamiento. El driver permite configurar una rampa de seis puntos donde los valores a encontrar, A1, AMAX, D1, DMAZ, V1 y VMAX corresponden a la aceleración, desaceleración y velocidad, como se observa en la figura 3.11.

Para el control de los movimientos se optó por generar una rampa de cuatro puntos. Se implementó la siguiente relación de variables:

- A1 = AMAX (Aceleración establecida por usuario).
- D1 = DMAZ (Desaceleración establecida por usuario).
- VMAX (Velocidad establecida por usuario).
- V1 = VMAX / 2 (Velocidad fija del movimiento).

el valor de stallguard2 disminuye. Se debe configurar entonces un valor límite de detección para que una vez alcanzado se genere un evento.

El firmware cuenta con una función llamada mod-coating-process-cero-machine que utiliza stallguard2. Se encarga de detectar el evento que el driver envía cuando se detecta un final de carrera. La misma se encarga de parar el motor, generar un desplazamiento en sentido contrario y establecer una nueva posición. Esta función se ejecuta al encender el equipo y sirve para posicionar el carro.

Otros parámetros importantes a configurar son los que definen la rampa de aceleración del desplazamiento. El driver permite configurar una rampa de seis puntos donde los valores a encontrar, A1, AMAX, D1, DMAX, V1 y VMAX corresponden a la aceleración, desaceleración y velocidad, como se observa en la figura 3.11.

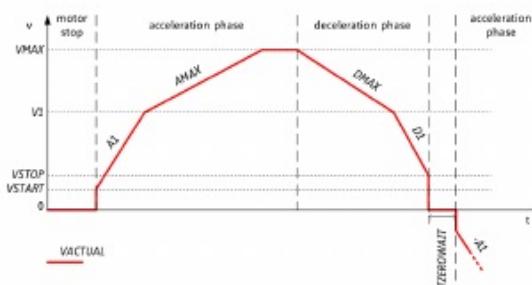


FIGURA 3.11. Configuración de rampa de seis puntos.

Para el control de los movimientos se optó por generar una rampa de cuatro puntos. Se implementó la siguiente relación de variables:

- A1 = AMAX (Aceleración establecida por usuario).
- D1 = DMAX (Desaceleración establecida por usuario).
- VMAX (Velocidad establecida por usuario).
- V1 = VMAX / 2 (Velocidad fija del movimiento).

Estas relaciones de variables están fijas en el firmware y no pueden ser cambiadas por el usuario, es decir que los movimientos del equipo siempre responden a una rampa de aceleración de cuatro puntos.

A modo de ejemplo, se presenta el fragmento de código 3.1 que pertenece a la aplicación app coating y se ejecuta cuando llega un comando de movimiento individual.

```

1 int HandlerDown_without_program(processCommandArgSpin_t* arg) {
2     int32_t reg_rampstat, position_actual , position_target;
3     processDipCoating.config.status=1;
4     Evalboards.ch1.enableDriver(DRIVER_ENABLE);
5
6     //Leo Posicion Actual
7     Evalboards.ch1.readRegister(0, 0x21, &position_actual);
8     // Set velocidad
9     //V1
10    Evalboards.ch1.writeRegister(0, TMC5130_V1, (arg->velocity) / 2);
11    //VMAX
12    Evalboards.ch1.writeRegister(0, TMC5130_VMAX, arg->velocity);
13    //VSTART
14    Evalboards.ch1.writeRegister(0, TMC5130_VSTART, 0);
15    //VSTOP

```

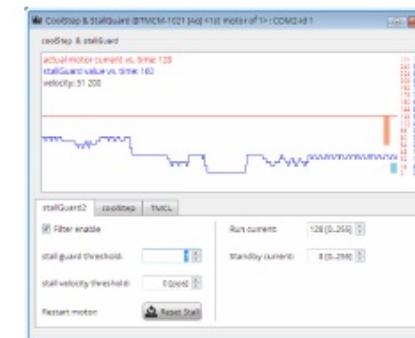


FIGURA 3.10. Configuración de funcionalidades stallguard2 y coolstep.

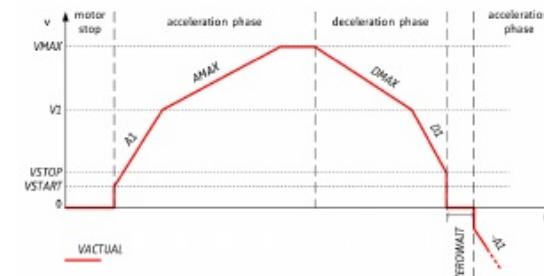


FIGURA 3.11. Configuración de rampa de seis puntos.

Estas relaciones de variables están fijas en el firmware y no pueden ser cambiadas por el usuario, es decir que los movimientos del equipo siempre responden a una rampa de aceleración de cuatro puntos.

A modo de ejemplo, se presenta el siguiente bloque de código que es parte de la app coating y se ejecuta cuando llega un comando de movimiento individual.

```

1 int HandlerDown_without_program(processCommandArgSpin_t* arg) {
2     int32_t reg_rampstat, position_actual , position_target;
3     processDipCoating.config.status=1;
4     Evalboards.ch1.enableDriver(DRIVER_ENABLE);
5
6     //Leo Posicion Actual
7     Evalboards.ch1.readRegister(0, 0x21, &position_actual);
8     // Set velocidad
9     //V1
10    Evalboards.ch1.writeRegister(0, TMC5130_V1, (arg->velocity) / 2);
11    //VMAX
12    Evalboards.ch1.writeRegister(0, TMC5130_VMAX, arg->velocity);
13    //VSTART
14    Evalboards.ch1.writeRegister(0, TMC5130_VSTART, 0);
15    //VSTOP

```

```

33 //VMAX
34 Evalboards.ch1.writeRegister(0, TMC5130_VMAX, arg->velocity);
35 //VSTART
36 Evalboards.ch1.writeRegister(0, TMC5130_VSTART, 0);
37 //VSTOP
38 Evalboards.ch1.writeRegister(0, TMC5130_VSTOP, 100);
39 // Set aceleracion y desaceleracion
40 //A1
41 Evalboards.ch1.writeRegister(0, TMC5130_A1, arg->acceleration);
42 //AMAX
43 Evalboards.ch1.writeRegister(0, TMC5130_AMAX, arg->acceleration);
44 // Seteo desaceleracion
45 //DMAX
46 Evalboards.ch1.writeRegister(0, TMC5130_DMAX, arg->acceleration);
47 //D1
48 Evalboards.ch1.writeRegister(0, TMC5130_D1, arg->acceleration);
49 /* Seteo de registro XTARGET*/
50 position_target=position_actual+arg->displacement_z;
51 position_target = mod_coating_handlers_control_limit_(position_target)
52 ;
53 Evalboards.ch1.writeRegister(0, TMC5130_XTARGET, position_target);
54 /*Detecto el flag que detecta XACTUAL=XTARGET y apago driver*/
55 while (1 == processDipCoating.config.status) {
56     Evalboards.ch1.readRegister (0, TMC5130_RAMPSTAT, &reg_rampstat);
57     /*Leo registro y comparo, velocidad zero y position_actual ==
58     position_target */
59     if (reg_rampstat & 0x00000600) {
60         break;
61     }
62     else {
63         vTaskDelay (OS_CONFIG_MOD_HANDLERS_COMMANDS_TASK_PERIOD /
64         portTICK_PERIOD_MS);
65     }
66 }
67 vTaskDelay (OS_CONFIG_MOD_HANDLERS_COMMANDS_TASK_PERIOD_LONG /
68 portTICK_PERIOD_MS);
69 processDipCoating.config.status=0;
70 Evalboards.ch1.enableDriver(DRIVER_DISABLE);
71
72 return 0;
73 }
```

CÓDIGO 3.1. Ejecución de comando DOWN.

Se detallan a continuación las líneas más importantes:

- 1 Recepción de estructura de datos con valores de aceleración, velocidad y desplazamiento.
- 4 Habilitación del driver.
- 7 Registro de la posición actual.
- 8-26 Carga de parámetros de velocidad y aceleración que forman rampa de cuatro puntos.
- 29 Suma el desplazamiento deseado con la posición actual.
- 30 Control de límites mecánicos.
- 31 Se escribe el registro XTARGET y se da inicio al movimiento.

```

16 Evalboards.ch1.writeRegister(0, TMC5130_VSTOP, 100);
17 // Set aceleracion y desaceleracion
18 //A1
19 Evalboards.ch1.writeRegister(0, TMC5130_A1, arg->acceleration);
20 //AMAX
21 Evalboards.ch1.writeRegister(0, TMC5130_AMAX, arg->acceleration);
22 // DMAX
23 Evalboards.ch1.writeRegister(0, TMC5130_DMAX, arg->acceleration);
24 //D1
25 Evalboards.ch1.writeRegister(0, TMC5130_D1, arg->acceleration);
26 /* Seteo de registro XTARGET*/
27 position_target=position_actual+arg->displacement_z;
28 position_target = mod_coating_handlers_control_limit_(position_target)
29 ;
30 Evalboards.ch1.writeRegister(0, TMC5130_XTARGET, position_target);
31 /*Detecto el flag que detecta XACTUAL=XTARGET y apago driver*/
32 while (1 == processDipCoating.config.status) {
33     Evalboards.ch1.readRegister (0, TMC5130_RAMPSTAT, &reg_rampstat);
34     /*Leo registro y comparo, velocidad zero y position_actual ==
35     position_target */
36     if (reg_rampstat & 0x00000600) {
37         break;
38     }
39     else {
40         vTaskDelay (OS_CONFIG_MOD_HANDLERS_COMMANDS_TASK_PERIOD /
41         portTICK_PERIOD_MS);
42     }
43 }
44 vTaskDelay (OS_CONFIG_MOD_HANDLERS_COMMANDS_TASK_PERIOD_LONG /
45 portTICK_PERIOD_MS);
46 processDipCoating.config.status=0;
47 Evalboards.ch1.enableDriver(DRIVER_DISABLE);
48
49 return 0;
50 }
```

CÓDIGO 3.1. Ejecución de comando DOWN.

Se detallan a continuación las líneas más importantes:

- 1 Recepción de estructura de datos con valores de aceleración, velocidad y desplazamiento.
- 4 Habilitación del driver.
- 7 Registro de la posición actual.
- 8-26 Carga de parámetros de velocidad y aceleración que forman rampa de cuatro puntos.
- 29 Suma el desplazamiento con la posición actual.
- 30 Control de límites mecánicos.
- 31 Se escribe el registro XTARGET y se da inicio al movimiento.
- 37 Dentro de un ciclo while se controlan dos bits del registro RAMPSTAT que se activan cuando XACTUAL = XTARGET (fin del movimiento).
- 46 Deshabilitación del driver si se cumple la condición anterior.

37 Dentro de un ciclo *while* se controlan dos bits del registro RAMPSTAT que se activan cuando XACTUAL = XTARGET (fin del movimiento).

46 Deshabilitación del driver si se cumple la condición anterior.

Otra parte importante de app coating es la definición del proceso completo de dip coating, el mismo está implementado con un arreglo de estructuras de tamaño fijo. Cada ítem del arreglo esta compuesto por el tipo de movimiento, los valores de parámetros que se reciben desde app consola o app hmi y punteros a funciones que ejecutan cada movimiento. Es decir que el proceso completo de dip coating está formado por una concatenación de movimientos individuales, como puede observarse en el siguiente fragmento de código 3.2. Cada vez que se ejecuta el proceso se recorre el arreglo.

```

1 processCommand_t cmdProcessCustom[MAX_JSTATIC_COMMAND] = {
2     /*Desplazamiento hasta muestras*/
3     { .cnum = COMMAND_DOWN, .arg.vel = 212000, .arg.accel = 65000, .fp =
4         DownUntil },
5     { .cnum = COMMAND_WAIT, .arg.time = 2, .fp = Wait },
6     /*Comienzo de ciclo*/
7     { .cnum = COMMAND_DOWN, .arg.vel = 100000, .arg.accel = 65000, .fp =
8         DownLoop },
9     { .cnum = COMMAND_WAIT, .arg.wait.time = 1, .fp = WaitDown },
10    { .cnum = COMMAND_UP, .arg.vel = 100000, .arg.accel = 65000, .fp =
11        UpLoop },
12    { .cnum = COMMAND_WAIT, .arg.time = 1, .fp = WaitUp },
13    /*Repeticiones del ciclo */
14    { .cnum = COMMAND_LOOP, .arg.value = 3 },
15    /*Fin de ciclo */
16    { .cnum = COMMAND_FINISH, .arg.vel = 0, .arg.accel = 0, .fp = Finish },
17 };

```

CÓDIGO 3.2. Proceso completo de dip coating.

Consola de comandos

App consola permite establecer un canal de comunicación entre el equipo dip coater y un ordenador a través de una comunicación serial. Como se mencionó en la subsección 3.1.1, la placa electrónica incorpora un conversor serie TTL-USB que permite conectar el equipo directamente a través de un cable USB.

Se definen dos series de comandos:

- Comandos para usuarios que permiten ejecutar un proceso completo de dip coating y movimientos individuales.
- Comandos para realizar consultas y configuraciones del sistema, utilizados durante el desarrollo del trabajo.

En la figura 3.12 se observa la primera serie de comandos.

Para realizar un proceso dip coating a través de la consola se debe seguir el siguiente procedimiento de ejecución de comandos:

1. Down, up, etc: permiten mover la muestra hasta la posición inicial del experimento.
2. Cerosample: registra la posición.

Otra parte importante de app coating es la definición del proceso completo de dip coating, el mismo está implementado con un arreglo de estructuras de tamaño fijo. Cada ítem del arreglo esta compuesto por tipo de movimiento, valores de parámetros que se reciben desde app consola o app hmi y punteros a funciones que ejecutan cada movimiento. Es decir que el proceso completo de dip coating está formado por una concatenación de movimientos individuales como puede observarse en el siguiente bloque de código.

```

1 processCommand_t cmdProcessCustom[MAX_JSTATIC_COMMAND] = {
2     /*Desplazamiento hasta muestras*/
3     { .commandnumber = PROCESS_COMMAND_DOWN, .fp = HandlerDownUntil },
4     { .commandnumber = PROCESS_COMMAND_WAIT, .fp = HandlerWait },
5     /*Comienzo de ciclo*/
6     { .commandnumber = PROCESS_COMMAND_DOWN, .fp = HandlerDownLoop },
7     { .commandnumber = PROCESS_COMMAND_WAIT, .fp = HandlerWaitDown },
8     { .commandnumber = PROCESS_COMMAND_UP, .fp = HandlerUpLoop },
9     { .commandnumber = PROCESS_COMMAND_WAIT, .fp = HandlerWaitUp },
10    /*Fin de ciclo */
11    { .commandnumber = PROCESS_COMMAND_FINISH, .fpcommandhandler =
12        HandlerFinish },
13 };

```

CÓDIGO 3.2. Proceso completo de dip coating.

Esta implementación es fija en el firmware pero se puede modificar con facilidad. Surgió en charlas con investigadores usuarios de equipos dip coater el interés por la posibilidad de modificar el proceso. Particularmente se observó interés en poder extraer la muestra con dos velocidades diferentes para generar en un mismo *films* dos perfiles diferentes.

Consola de comandos

App consola permite establecer un canal de comunicación entre el equipo dip coater y un ordenador a través de una comunicación serial. Como se mencionó en la subsección 3.1.1, la placa electrónica incorpora un conversor serie TTL-USB que permite conectar el equipo directamente a través de un cable USB.

Se definen dos series de comandos:

- Comandos para usuarios que permiten ejecutar un proceso completo de dip coating y movimientos individuales.
- Comandos para realizar consultas y configuraciones del sistema, utilizados durante el desarrollo del trabajo.

En la figura 3.12 se observa la primera serie de comandos.

Para realizar un proceso dip coating a través de la consola se debe seguir el siguiente procedimiento de ejecución de comandos:

1. Down, up, etc: permiten mover la muestra hasta la posición inicial del experimento.
2. Cerosample: registra la posición.
3. Depthsample: configura la distancia de recorrido de la muestra.
4. Setcommandcustomapp: configura los parámetros del proceso listados a continuación:

```

setcommandcustomapp [-d=<n>] [-da=<n>] [-du=<n>] [-w=<n>] [-uw=<n>] [-uus=<n>] [-l=<n>]
  Dip Coating Set Program Custom
    -d=<n> Down Velocity
    -da=<n> Down Acceleration
    -du=<n> Down Wait_ms
    -w=<n> Up Velocity
    -uw=<n> Up Acceleration
    -uus=<n> Up Wait_ms
    -l=<n> Total Loop

zerosample
  Dip Coating Program Set Zero of sample

depthsample [-d <n>]
  Dip Coating Program Set Delta Depth Sample
    -d <n> Delta Depth Sample [xx]

camerachine
  Dip Coating Set Camo Machine

run
  Dip Coating Start

downfast
  Dip Coating Down Fast

down
  Dip Coating Down

downlow
  Dip Coating Down Fast

upfast
  Dip Coating Up Fast

up
  Dip Coating Up

upslow
  Dip Coating Up Slow

stop
  Dip Coating Emergency Stop

```

FIGURA 3.12. Comandos de movimientos.

3. Depthsample: configura la distancia de recorrido de la muestra.
4. Setcommandcustomapp: configura la velocidad y aceleración ascendente y descendente, los tiempos de espera en posición superior e inferior y la cantidad de repeticiones del ciclo.
5. Run: inicia el proceso dip coating.
6. Stop: está disponible para que el usuario pueda detener el proceso.

Se observa en la figura 3.13 la segunda serie de comandos.

Estos comandos permiten utilizar al driver de manera independiente, realizar configuraciones y visualizar el estado de los registros. Los comandos read-register y write-register sirven para leer y escribir registros sobre el driver TMC5130.

A modo de ejemplo se observa en la figura 3.14 una consulta sobre el registro Xactual[0x21], que expresa la posición actual en **micropasos** desde la **referencia inicial** y otra consulta sobre el registro Xtarg[0x2D], cuyo valor expresa la **posición objetivo** en micropasos que se desea alcanzar. En este ejemplo Xactual = Xtarg ya que el carro estaba detenido. Para ejecutar un movimiento se debe configurar una posición en Xtarg, la misma accionará el motor hasta alcanzar

```

setcommandcustomapp [-d=<n>] [-da=<n>] [-du=<n>] [-w=<n>] [-uw=<n>] [-uus=<n>] [-l=<n>]
  Dip Coating Set Program Custom
    -d=<n> Down Velocity
    -da=<n> Down Acceleration
    -du=<n> Down Wait_ms
    -w=<n> Up Velocity
    -uw=<n> Up Acceleration
    -uus=<n> Up Wait_ms
    -l=<n> Total Loop

zerosample
  Dip Coating Program Set Zero of sample

depthsample [-d <n>]
  Dip Coating Program Set Delta Depth Sample
    -d <n> Delta Depth Sample [xx]

camerachine
  Dip Coating Set Camo Machine

run
  Dip Coating Start

downfast
  Dip Coating Down Fast

down
  Dip Coating Down

downlow
  Dip Coating Down Fast

upfast
  Dip Coating Up Fast

up
  Dip Coating Up

upslow
  Dip Coating Up Slow

stop
  Dip Coating Emergency Stop

```

FIGURA 3.12. Comandos de movimientos.

- **Velocidad y aceleración ascendente y descendente.**
 - **Tiempos de espera en posición superior e inferior.**
 - **Cantidad de repeticiones del ciclo.**
5. Run: inicia el proceso dip coating.
 6. Stop: está disponible para que el usuario pueda detener el proceso si así lo deseará.

Se observa en la figura 3.13 la segunda serie de comandos.

Estos comandos permiten utilizar al driver de manera independiente, realizar configuraciones y visualizar el estado de los registros. Los comandos read-register y write-register sirven para leer y escribir registros sobre el driver TMC5130.

A modo de ejemplo se observa en la figura 3.14 una consulta sobre el registro Xactual[0x21], que expresa la posición actual en **micropasos** desde la **referencia inicial** y otra consulta sobre el registro Xtarg[0x2D], cuyo valor expresa la **posición objetivo** en **micropasos** que se desea alcanzar. En este ejemplo Xactual = Xtarg ya que el carro estaba detenido. Para ejecutar un movimiento se debe configurar una posición en Xtarg, la misma accionará el motor hasta alcanzar

```

data
Dip Coating Read all data saved

ena
Dip Coating General Enable Driver TMC5130

dis
Dip Coating General Disable Driver TMC5130

reset
Dip Coating Reset

timestamp
Dip Coating System Time

read_register_tmc [-a <n>]
Read TMC Register
-a, --address to write=<n> Write TMC Register [HEX]
-v, --value=<n> Write TMC Register [HEX]

write_register_tmc [-a <n>] [-v <n>]
Write TMC Register
-a, --address to write=<n> Write TMC Register [HEX]
-v, --value=<n> Write TMC Register [HEX]

```

FIGURA 3.13. Comandos de control.

Xactual = Xtarget. Previamente se deben configurar los registros de velocidad y aceleración.

```

tecscl_dipcoater> read_register_tmc -a 0x21
I (1572337) mod_console_commands: Address->0x21 Value->0x00108e95
tecscl_dipcoater> read_register_tmc -a 0x20
I (1580067) mod_console_commands: Address->0x2d Value->0x00108e95
tecscl_dipcoater> 

```

FIGURA 3.14. Lectura de registros del driver TMC5130.

Pantalla táctil

La aplicación app hmi establece un canal de comunicación serial entre la pantalla táctil y el microcontrolador, se encarga de procesar datagramas salientes y entrantes.

En la sección 2.3 se presentó el modelo STWI043WT elegido para el equipo dip coater, requiere para su configuración el desarrollo de un proyecto con el software *STONE GUI Desing Software*. La interfaz interactiva permite crear pantallas y diferentes tipos de objetos o widgets para brindar de funcionalidad a las mismas. El fabricante define también un protocolo de comunicación [30] para interactuar con los objetos que componen cada pantalla.

En la figura 3.16 se define el formato de datagrama para enviar datos desde el microcontrolador hacia la pantalla.

El datagrama está compuesto por tres bloques:

1. Frame header: 3 bytes fijos.
2. Data: Datos definidos en texto plano con formato *JSON (JavaScript Object Notation)*.

```

data
Dip Coating Read all data saved

ena
Dip Coating General Enable Driver TMC5130

dis
Dip Coating General Disable Driver TMC5130

reset
Dip Coating Reset

timestamp
Dip Coating System Time

read_register_tmc [-a <n>]
Read TMC Register
-a, --address to write=<n> Write TMC Register [HEX]
-v, --value=<n> Write TMC Register [HEX]

write_register_tmc [-a <n>] [-v <n>]
Write TMC Register
-a, --address to write=<n> Write TMC Register [HEX]
-v, --value=<n> Write TMC Register [HEX]

```

FIGURA 3.13. Comandos de control.

Xactual = Xtarget. Previamente se deben configurar los registros de velocidad y aceleración.

```

tecscl_dipcoater> read_register_tmc -a 0x21
I (1572337) mod_console_commands: Address->0x21 Value->0x00108e95
tecscl_dipcoater> read_register_tmc -a 0x20
I (1580067) mod_console_commands: Address->0x2d Value->0x00108e95
tecscl_dipcoater> 

```

FIGURA 3.14. Lectura de registros del driver TMC5130.

Pantalla táctil

La aplicación app hmi establece un canal de comunicación serial entre la pantalla táctil y el microcontrolador, la misma se encarga de procesar datagramas salientes y entrantes.

En la sección 2.3 se presentó el modelo STWI043WT elegido para el equipo dip coater, el mismo requiere para su configuración el desarrollo de un proyecto con el software *STONE GUI Desing Software*. La interfaz interactiva permite crear pantallas y diferentes tipos de objetos o widgets para brindar de funcionalidad a las mismas. El fabricante define también un protocolo de comunicación [32] para interactuar con los objetos que componen cada pantalla.

En la figura 3.16 se define el formato de datagrama para enviar datos desde el microcontrolador hacia la pantalla.

El datagrama está compuesto por tres bloques:

1. Frame header: 3 bytes fijos.
2. Data: Datos definidos en texto plano con formato *JSON (JavaScript Object Notation)*.

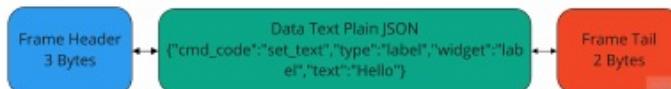


FIGURA 3.15. Datagrama desde microcontrolador hacia pantalla.

3. Frame tail: 2 bytes fijos.

El campo data define múltiples categorías, se mencionan a continuación las más importantes:

- Cmd-code: Es un identificador único que define la instrucción.
- Type: Define el tipo de objeto.
- Widget: Define el nombre único del objeto.
- Text: Define el contenido del objeto, varía según el tipo de objeto.

En la figura 3.16 se define el formato de datagrama para enviar datos desde la pantalla hacia el microcontrolador. El datagrama está compuesto por los siguientes bloques:

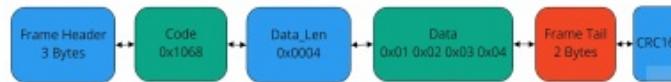


FIGURA 3.16. Datagrama desde pantalla hacia microcontrolador.

1. Frame header: 3 bytes fijos.
2. Code: Identificación única de objeto.
3. Data-Len: Define el largo del dato a transmitir.
4. Data: Su tamaño debe coincidir con el ítem anterior.
5. Frame tail: 2 bytes fijos.
6. CRC16: Para verificación de integridad del datagrama.

Para procesar los datagramas entrantes se implementó un bloque de código que analiza el buffer del periférico UART-1. Cuando hay datos disponibles el periférico envía un evento a través de una **cola** de freeRTOS. El evento **UART-DATA** se recibe en la **tarea** mod-hmi-RX-task-loop y comienza a procesar dichos **datos**. En la figura 3.17 se observa el control que se realiza en cada uno de los bytes **entrantes**. Si el datagrama pasa todas las **condiciones** se **acepta** y se **envia** a la **tarea** app hmi task a través de una **cola** para ser procesado.

En la figura 3.18 se presenta la pantalla de configuración de programa creada con el software de diseño.

Al ejecutar el botón **play**, la pantalla envía un serie de datagramas con todos los parámetros configurados en la **misma**. Los datagramas son procesados para verificar su validez y enviados hacia app coating para dar inicio al proceso.

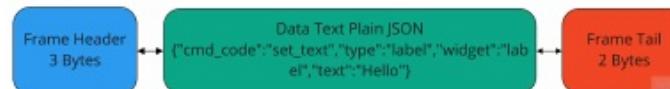


FIGURA 3.15. Datagrama desde microcontrolador hacia pantalla.

3. Frame tail: 2 bytes fijos.

El campo data define múltiples categorías, se mencionan a continuación las más importantes:

- Cmd-code: Es un identificador único que define la instrucción.
- Type: Define el tipo de objeto.
- Widget: Define el nombre único del objeto.
- Text: Define el contenido del objeto, varía según el tipo de objeto.

En la figura 3.16 se define el formato de datagrama para enviar datos desde la pantalla hacia el microcontrolador. El datagrama está compuesto por los siguientes bloques:



FIGURA 3.16. Datagrama desde pantalla hacia microcontrolador.

1. Frame header: 3 bytes fijos.
2. Code: Identificación única de objeto.
3. Data-Len: Define el largo del dato a transmitir.
4. Data: Su tamaño debe coincidir con el ítem anterior.
5. Frame tail: 2 bytes fijos.
6. CRC16: Para verificación de integridad del datagrama.

Para procesar los datagramas entrantes se implementó un bloque de código que analiza el buffer del periférico UART-1. Cuando hay datos disponibles el periférico envía un evento a través de una **queue** de freeRTOS. El evento **UART-DATA** se recibe en la **task** mod-hmi-RX-task-loop la cual comienza a procesar dichos **datos** mientras estén disponibles. En la figura 3.17 se observa el control que se va realizando a cada uno de los bytes entrantes. Si el datagrama pasa todas las **condiciones** es **aceptado** y **enviado** a la **task** app hmi task a través de una **queue** para ser procesado.

En la figura 3.18 se presenta la pantalla de configuración de programa creada con el software de diseño.

Al ejecutar el botón **play**, la pantalla envía un serie de datagramas con todos los parámetros configurados en la **misma**, los datagramas son procesados para verificar su validez y enviados hacia app coating para dar inicio al proceso.

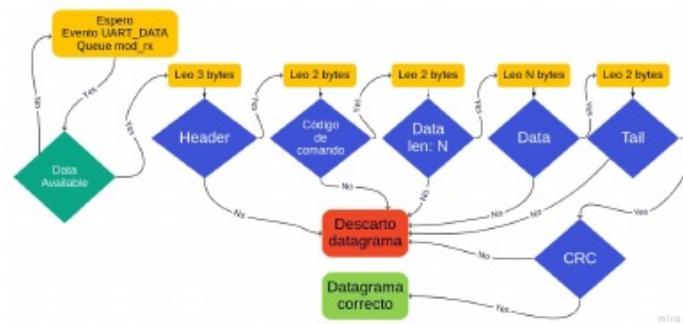


FIGURA 3.17. Secuencia de procesamientos de datos entrantes.



FIGURA 3.18. Pantalla de configuración de programa.

Registro de variables ambientales

En la sección 2.5 se presentó el requerimiento opcional que establece el registro de variables de humedad, presión y temperatura. El interés del cliente se fundamenta en que ciertos experimentos necesitan realizarse a humedad y temperatura controlada. Esta primera implementación solo monitorea las variables y no realiza ningún control de corrección sobre las mismas.

Para el registro de estas variables se incorporó al sistema un sensor BME280. El mismo integra en un solo CI sensores de presión atmosférica, temperatura y humedad relativa. El fabricante ofrece en sus repositorios [31] ejemplos de implementaciones y una API para utilizar todas las funcionalidades del integrado.

La comunicación entre el sensor y el microcontrolador se realizó a través del protocolo I2C. Se incorporó el módulo de software BOARD I2C que se encarga de inicializar el periférico I2C. También se implementó el módulo API BOSH BME, que trabaja con todas las funcionalidades que ofrece el fabricante en su API. En la figura 3.19 se puede observar el detalle de las capas desarrolladas.

API BOSH BME inicializa una estructura que define los siguientes parámetros:

- Dirección del dispositivo I2C.

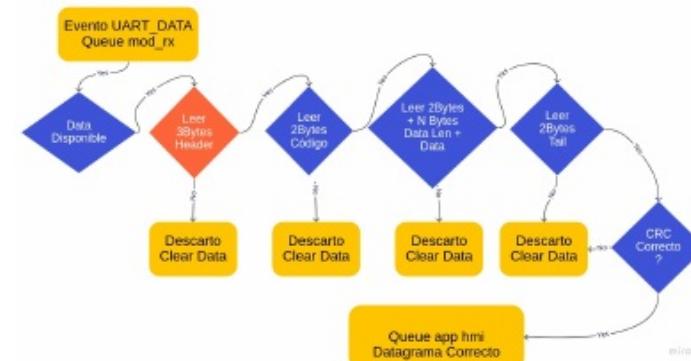


FIGURA 3.17. Secuencia de procesamientos de datos entrantes.



FIGURA 3.18. Pantalla de configuración de programa.

Registros de variables ambientales

En la sección 2.5 se presentó el requerimiento opcional que establece el registro de variables de humedad, presión y temperatura. El interés del cliente se fundamenta en que ciertos experimentos necesitan realizarse particularmente a humedad y temperatura controlada.

Para el registro de estas variables se incorporó al sistema un sensor BME280, el mismo integra en un solo CI sensores de presión atmosférica, temperatura y humedad relativa. El fabricante ofrece en sus repositorios [33] ejemplos de implementaciones y una API para utilizar todas las funcionalidades del CI.

La comunicación entre el sensor y el microcontrolador se realizó a través del protocolo I2C, se incorporó el módulo del software BOARD I2C que se encarga de inicializar el periférico. También se implementó el módulo API BOSH BME, que trabaja con todas las funcionalidades que ofrece el fabricante en su API. En la figura 3.19 de observa el flujo de capas desarrollado:

API BOSH BME inicializada una estructura donde define los siguientes parámetros de interés:

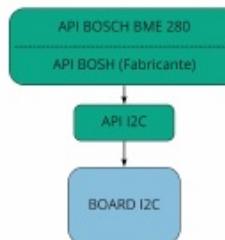


FIGURA 3.19. Módulo API BOSH.

- Función de lectura.
- Función de escritura.
- Función de delay.
- Configuración de modo normal de funcionamiento del sensor.

Los datos registrados por el sensor se visualizan cada cinco minutos en la consola de comandos, como se observa en la figura 3.20.

```

TECSCI_SAS      ESP-IDF Console,
Type 'help' to get the list of commands.
Use UP/DOWN arrows to navigate through command history.
Press TAB when typing command name to auto-complete,
Press Ctrl+C will terminate the console environment.

 tecsci_dipcoater>
I (2507) esp_bosch_bme280.ct Temperatura:22.85, Presion:100811.02, Humedad:48.56
  
```

FIGURA 3.20. Registro de datos en consola.

Parámetros de calibración

La carpeta /components/config contiene tres archivos de configuración importantes:

- hardware.h: contiene todas las macros referidas a los pines de conexión del modelo de microcontrolador utilizado.
- os-config.h: contiene las macros de configuración de las tareas y colas de FreeRTOS. Incluye entre otros parámetros el tamaño del stack, la prioridad y el tiempo de ejecución de cada tarea.
- machine.h: contiene las macros relacionadas con la calibración mecánica del equipo.

La macro más importante configurada en el archivo machine.h es MACHINE STEPS PER MILLIMETER, que debe estar perfectamente determinada. En la sección 4.3 se demuestra el procedimiento realizado para obtener su valor. Esta macro define la cantidad de micropasos necesarios para generar el desplazamiento de 1 mm. La misma está completamente relacionada con el paso del tornillo acoplado al eje del motor. Las unidades de posición son expresados en micropasos,

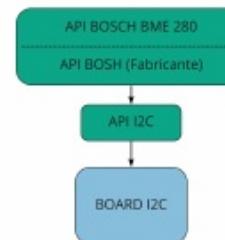


FIGURA 3.19. Módulo API BOSH.

- Dirección del dispositivo I2C.
- Función de lectura.
- Función de escritura.
- Función de delay.
- Configuración de modo normal de funcionamiento del sensor.

Para visualizar los datos registrados por el sensor se implementó sobre la APP TEST una rutina de consulta con llamado a funciones de la capa API BOSH BME, se observa en la siguiente figura 3.20 los datos registrados en consola.

```

TECSCI_SAS      ESP-IDF Console,
Type 'help' to get the list of commands,
Use UP/DOWN arrows to navigate through command history.
Press TAB when typing command name to auto-complete,
Press Ctrl+C will terminate the console environment.

 tecsci_dipcoater>
I (2507) esp_bosch_bme280.ct Temperatura:22.85, Presion:100811.02, Humedad:48.56
  
```

FIGURA 3.20. Registro de datos en consola.

El desarrollo de este MPV incluyó el registro de variables ambientales pero no un control y corrección de las mismas con un sistema de control. Sin embargo, se pretende ofrecer en un futuro cercano la funcionalidad de cámara de humedad anexada a este MPV.

Parámetros de calibración

La carpeta /components/config contiene tres archivos de configuración importantes:

- hardware.h: contiene todas las macros referidas a los pines de conexión del modelo de microcontrolador utilizado.
- os-config.h: contiene las macros de configuración de las tareas y colas de FreeRTOS. Entre otras los tamaños de stack, las prioridades y los períodos de tiempo de ejecución de tareas.

las unidades de velocidad en **micropasos** sobre segundos y las unidades de **aceleración en micropasos** sobre segundos al cuadrado.

Se observa en la figura 3.21, extraída de la hoja de datos del driver TMC5130, los factores de corrección que deben aplicarse cuando se cuenta con un clock externo incorporado en el circuito electrónico.

PARAMETER VS. UNITS		
Parameter / Symbol	Unit	calculation / description / comment
f _{0,0} [Hz]	[Hz]	clock frequency of the TMC5130A in [Hz]
s	[s]	second
US	ustep	
FS	fullstep	
ustep velocity v[Hz]	usteps / s	v[Hz] = v[5130A] * (f _{0,0} [Hz]/2 / 2 ²³)
ustep acceleration a[Hz/s]	usteps / s ²	a[Hz/s] = a[5130A] * f _{0,0} [Hz] ² / (512*256) / 2 ²⁴

FIGURA 3.21. Unidades.

Se observa en el fragmento de código 3.3 el cálculo de las constantes que corrigen los valores de velocidad y aceleración.

```

1 #define MACHINE_STEPS_PER_MILLIMETER (12916)
2 #define MACHINE_EXT_CLOCK (16000000) //16MHz
3
4
5
6 /*FACTOR*/
7 /*((MACHINE_EXT_CLOCK/2)*(1/8388608))*/
8 #define MACHINE_USTEPS_VELOCITY_FACTOR (0.9536743164)
9 /*((MACHINE_EXT_CLOCK*MACHINE_EXT_CLOCK)/(512*256)/(16777216))*/
10 #define MACHINE_USTEPS_ACCELERATION_FACTOR (116.4153218)
11
12
13 /*UPPER AND LOWER MECHANICAL LIMIT*/
14
15 #define UPPER_LIMIT (MACHINE_STEPS_PER_MILLIMETER * 10) // 10mm
16 #define LOWER_LIMIT (MACHINE_STEPS_PER_MILLIMETER * 290) // 290mm

```

CÓDIGO 3.3. Macros de desplazamiento y factores de corrección.

3.3. Estructura mecánica

3.3.1. Fabricación de piezas personalizadas a través de mecanizado CNC

Etapa CAD

Como se mencionó en la sección 2.4 se utilizó para el diseño mecánico del equipo el software BOBCAD. El módulo CAD del software permite realizar modelos 2D y 3D de piezas.

El prototipo dip coater cuenta actualmente con dos piezas mecanizadas en aluminio. Se observa en la figura 3.22 la pieza que se acopla al carro de la guía lineal presentada en la sección 2.4.

Y en la figura 3.23 el soporte superior que sostiene el motor paso a paso y el tornillo acoplado al eje del motor.

Con los modelos 3D terminados, se fabricó un primer lote utilizando impresión 3D en plástico. Luego que las piezas fueron probadas, testeadas y aprobadas en el prototipo, se pasó a la fabricación final sobre aluminio.

- machine.h: contiene las macros relacionadas con la calibración mecánica del equipo.

La macro más importante configurada en el archivo machine.h es MACHINE STEPS PER MILLIMETER y es necesario que este bien definida. En la sección 4.3 se demuestra el procedimiento realizado para definir su valor. Esta macro define la cantidad de micro pasos necesarios para generar el desplazamiento de 1 mm. La misma esta completamente relaciona con el paso del tornillo acoplado al eje del motor. Las unidades de posición son expresados en micro pasos, las unidades de velocidad en micro pasos sobre segundos y las unidades de aceleración en micro pasos sobre segundos al cuadrado.

Se observa en la figura 3.21, extraída de la hoja de datos del driver TMC5130, los factores de corrección que deben aplicarse cuando se cuenta con un clock externo incorporado en el circuito electrónico. Se deben calcular las constantes que corrigen los valores de velocidad y aceleración.

PARAMETER VS. UNITS		
Parameter / Symbol	Unit	calculation / description / comment
f _{0,0} [Hz]	[Hz]	clock frequency of the TMC5130A in [Hz]
s	[s]	second
US	ustep	
FS	fullstep	
ustep velocity v[Hz]	usteps / s	v[Hz] = v[5130A] * (f _{0,0} [Hz]/2 / 2 ²³)
ustep acceleration a[Hz/s]	usteps / s ²	a[Hz/s] = a[5130A] * f _{0,0} [Hz] ² / (512*256) / 2 ²⁴

FIGURA 3.21. Unidades.

Como se mencionó en la subsección 2.2.1 el equipo se configuró con 51200 micro pasos por vuelta completa.

```

1
2 #define MACHINE_STEPS_PER_MILLIMETER (12916)
3 #define MACHINE_EXT_CLOCK (16000000) //16MHz
4
5
6 /*FACTOR*/
7 /*((MACHINE_EXT_CLOCK/2)*(1/8388608))*/
8 #define MACHINE_USTEPS_VELOCITY_FACTOR (0.9536743164)
9 /*((MACHINE_EXT_CLOCK*MACHINE_EXT_CLOCK)/(512*256)/(16777216))*/
10 #define MACHINE_USTEPS_ACCELERATION_FACTOR (116.4153218)
11
12
13 /*UPPER AND LOWER MECHANICAL LIMIT*/
14
15 #define MACHINE_CONTROL_MECHANICAL_UPPER_LIMIT (
16   MACHINE_STEPS_PER_MILLIMETER * 10) // 10mm
17 #define MACHINE_CONTROL_MECHANICAL_LOWER_LIMIT (
18   MACHINE_STEPS_PER_MILLIMETER * 290) // 290mm

```

CÓDIGO 3.3. Macros de desplazamiento y factores de conversión.

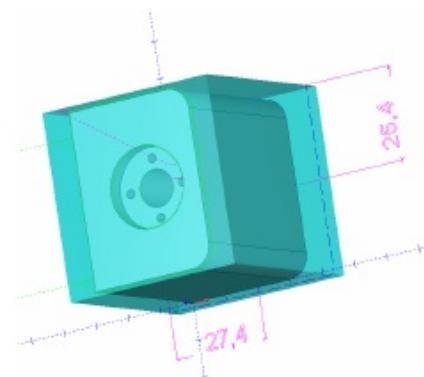


FIGURA 3.22. Pieza personalizada soporte de carro.

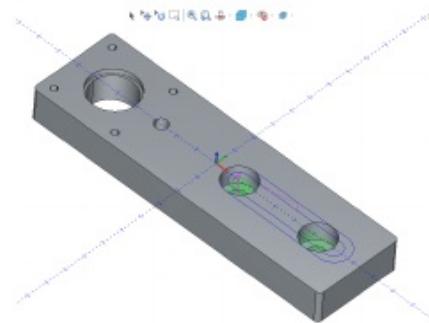


FIGURA 3.23. Piezas personalizada soporte de estructura superior.

Etapa CAM

La estrategia utilizada en el mecanizado CNC es el método de arranque de víspera. Este consiste en partir de un bloque de aluminio con volumen de material suficiente y desbastar con herramientas de corte hasta modelar la pieza. Esta estrategia se programa en la parte CAM del software. Existen diferentes operaciones de mecanizado que se utilizan según el tipo de pieza que se deseé fabricar. Tal es el caso de refrentado, vaciado, fresado de chaflán, taladrado y rosado entre otras. Cada una de estas operaciones en general se realizan con herramientas específicas que son definidas en la configuración del software.

A modo de ejemplo se presenta en la figura 3.24 el listado de operaciones realizadas para la fabricación de la pieza soporte carro del equipo.

En general las piezas se fabrican en dos etapas, primero a través de diferentes operaciones se mecaniza la parte superior, luego se rota la pieza 180° y se mecaniza la parte inferior.

3.3. Estructura mecánica**3.3.1. Fabricación de piezas personalizadas a través de mecanizado CNC****Etapa CAD**

Como se mencionó en la sección 2.4 se utilizó para el diseño mecánico del equipo el software BOBCAD. El módulo CAD del software permite realizar modelos 2D y 3D de piezas.

El prototipo dip coater cuenta actualmente con dos piezas mecanizadas en aluminio. Se observa en la figura 3.22 la pieza que se acopla al carro de la guía lineal presentada en la sección 2.4.

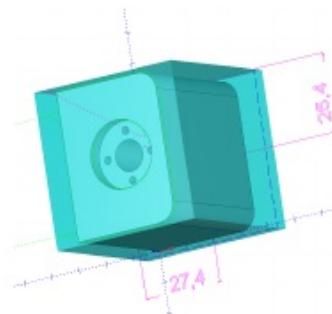


FIGURA 3.22. Pieza personalizada soporte de carro.

Y en la figura 3.23 el soporte superior que sostiene el motor paso a paso y el tornillo acoplado al eje del motor.

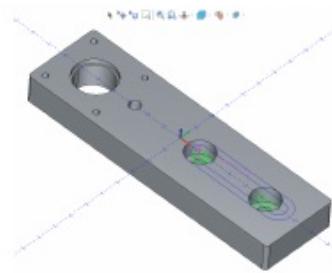


FIGURA 3.23. Piezas personalizada soporte de estructura superior.

Con los modelos 3D diseñados, se fabricaron primeras versiones a través de impresión 3D en plástico. Luego que las piezas fueron probadas, testeadas y aprobadas en el prototipo, se pasó a la fabricación final sobre aluminio.



FIGURA 3.24. Operaciones de mecanizado en software Bodcad.

El material mecanizado para la fabricación de estas piezas fue aluminio 6061, el mismo es una aleación endurecida compuesta por aluminio, magnesio y silicio. La elección se basó en que el mismo pue someterse a tratamientos de anodizado posteriores. El anodizado es un tratamiento electrolítico, que genera una capa superficial de óxido de aluminio (alúmina). De espesor superior que el aluminio en estado natural, tiene como ventajas la protección contra atmósferas agresivas, agentes químicos y una mayor dureza superficial.

Finalmente en la figura 3.25 se observan ambas piezas fabricadas.

3.3.2. Modelos 3D y real

En la figura 3.26 se presenta el primer modelo 3D diseñado del equipo dip coater.

Luego de sucesivas iteraciones con pruebas de piezas impresas en material plástico se logró fabricar un primer prototipo completamente en metal que se presenta en la figura 3.27.

Etapa CAM

La estrategia utilizada en el mecanizado CNC es por el método de arranque de viruta. Este consiste en partir de un bloque de aluminio con volumen de material suficiente y desbastar con herramientas de corte hasta modelar la pieza. Esta estrategia se programa en la parte CAM del software. Existen diferentes operaciones de mecanizado que se utilizan según el tipo de pieza que se deseé fabricar. Tal es el caso de refrentado, vaciado, fresado de chaflán, taladrado y rosado entre otras. Cada una de estas operaciones en general se realizan con herramientas específicas que son definidas en la configuración del software.

A modo de ejemplo se presenta en la figura 3.24 el listado de operaciones realizadas para la fabricación de la pieza soporte carro del equipo.

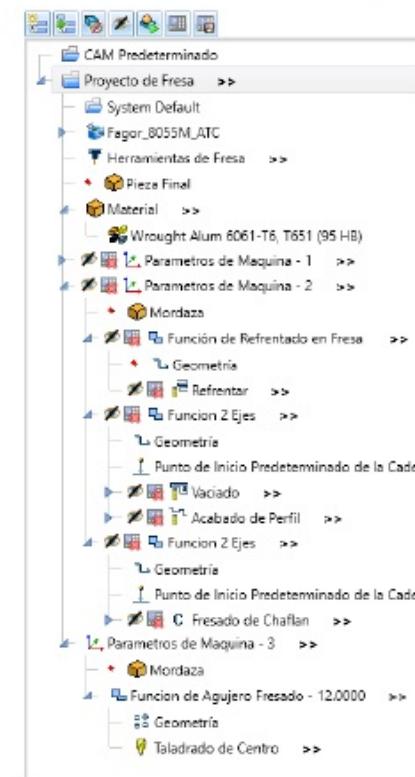


FIGURA 3.24. Operaciones de mecanizado en software Bodcad.

En general las piezas se fabrican en dos etapas, primero a través de diferentes operaciones se mecaniza la parte superior, luego se rota la pieza 180° y se mecaniza la parte inferior.

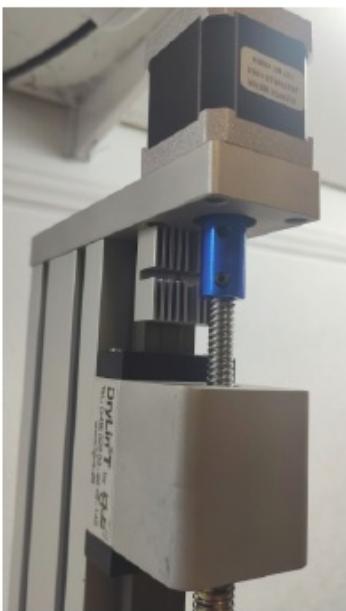


FIGURA 3.25. Piezas fabricadas en centro de mecanizado.

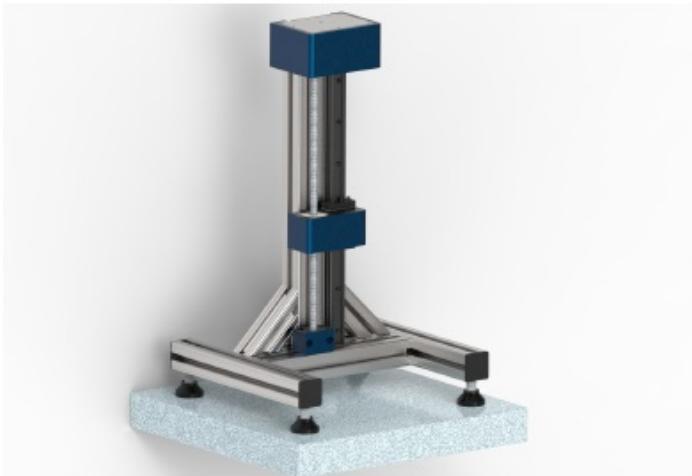


FIGURA 3.26. Modelo 3D.

El material mecanizado para la fabricación de estas piezas fue aluminio 6061, el mismo es una aleación endurecida compuesta por aluminio, magnesio y silicio. La elección se basó en que el mismo puede someterse a tratamientos de anodizado posteriores. El anodizado es un tratamiento electrolítico, que genera una capa superficial de óxido de aluminio (alúmina). De espesor superior que el aluminio en estado natural, tiene como ventajas la protección contra atmósferas agresivas, agentes químicos y una mayor dureza superficial.

Finalmente en la figura 3.25 se observan ambas piezas fabricadas.

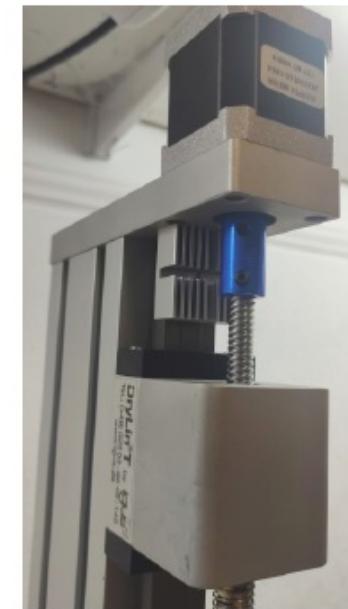


FIGURA 3.25. Piezas fabricadas en centro de mecanizado.

3.3.2. Modelos 3D y real

En la figura 3.26 se presenta el primer modelo 3D diseñado del equipo dip coater.

Luego de sucesivas iteraciones con pruebas de piezas impresas en material plástico se logró fabricar un primer prototipo completamente en metal que se presenta en la figura 3.27.

3.3. Estructura mecánica

37



FIGURA 3.27. Primer prototipo dip coater TECSCI.

3.3. Estructura mecánica

37



FIGURA 3.26. Modelo 3D.



FIGURA 3.27. Primer prototipo dip coater TECSCI.

4.2. Pruebas funcionales del firmware

41

En la figura 4.4 se observa la ejecución del comando de escritura sobre el registro [0x2D] con valor [0x00 0xFF 0xFF 0x00].

- MOSI: [0x2D + 0x80 = 0xAD][0x00 0xFF 0xFF 0x00].



FIGURA 4.4. Comando de escritura sobre registro [0x2D].

En la figura 4.5 se observa la ejecución nuevamente del comando de lectura sobre el registro [0x2D].

- MOSI: [0x2D][0x00 0xFF 0x00 0x00].
- MISO: [0x11][0x00 0xFF 0x00 0x00].



FIGURA 4.5. Comando de lectura actualizado sobre registro [0x2D].

Se observa entonces que luego de estas operaciones el registro [0x2D] se actualizó correctamente con el valor [0x00 0xFF 0x00 0x00].

Con este ensayo se validó la comunicación SPI entre el microcontrolador y el CI TMC5130 para las operaciones de lectura y escritura de datos.

4.2. Pruebas funcionales del firmware

4.2.1. Tiempo de ejecución de movimientos

El ensayo se realizó para verificar los parámetros que definen el desplazamiento de la muestra, es decir para verificar que las velocidades y aceleraciones que definen movimientos sean similares a las que surgen del cálculo teórico.

En el capítulo 3 se detalló la configuración de la rampa de cuatro puntos que define los movimientos del equipo. Con los valores de aceleración, desaceleración, velocidad y desplazamiento se calculó el tiempo teórico necesario para ejecutar cada ciclo de trabajo.

Para realizar el ensayo, cuyos parámetros se ven en la tabla 4.1, se implementó una aplicación de prueba que realiza el siguiente procedimiento:

4.2. Pruebas funcionales del firmware

41

En la figura 4.4 se observa la ejecución del comando de escritura sobre el registro [0x2D] con valor [0x00 0xFF 0xFF 0x00].

- MOSI: [0x2D + 0x80 = 0xAD][0x00 0xFF 0xFF 0x00].



FIGURA 4.4. Comando de escritura sobre registro [0x2D].

En la figura 4.5 se observa la ejecución nuevamente del comando de lectura sobre el registro [0x2D].

- MOSI: [0x2D][0x00 0xFF 0x00 0x00].
- MISO: [0x11][0x00 0xFF 0x00 0x00].



FIGURA 4.5. Comando de lectura actualizado sobre registro [0x2D].

Se observa entonces que luego de estas operaciones el registro [0x2D] se actualizó correctamente con el valor [0x00 0xFF 0x00 0x00].

Con este ensayo se validó la comunicación SPI entre el microcontrolador y el CI TMC5130 para las operaciones de lectura y escritura de datos.

4.2. Pruebas funcionales del firmware

4.2.1. Tiempo de ejecución de movimientos

El ensayo se realizó para verificar los parámetros que definen el desplazamiento de la muestra, es decir para verificar que las velocidades y aceleraciones que definen movimientos sean similares a las que surgen del cálculo teórico.

En el capítulo 3 se detalló la configuración de la rampa de cuatro puntos que define los movimientos del equipo. Con los valores de aceleración, desaceleración, velocidad y desplazamiento se calculó el tiempo teórico necesario para la ejecución de cada movimiento.

En la tabla 4.1 se observan los valores de los parámetros ensayados.

Para realizar el ensayo se implementó una aplicación de prueba que realiza el siguiente procedimiento:

TABLA 4.1. Ensayo de tiempos en desplazamientos

Velocidad (mm/min)	Aceleración-Desaceleración (m/min ²)	Desplazamiento (mm)
1	100-500-1000-2100	50
10	100-500-1000-2100	50
100	100-500-1000-2100	50
200	100-500-1000-2100	50
500	100-500-1000-2100	50
800	100-500-1000-2100	50

- Configuración de movimiento descendente con valores de velocidad, aceleración y desplazamiento.
- Ejecución del movimiento descendente y registro del tiempo del sistema.
- Registro del tiempo del sistema al final del movimiento, cálculo de variación temporal y envío del dato por terminal serie.
- Configuración y ejecución de movimiento ascendente con valores de velocidad, aceleración y desplazamiento.
- Ejecución del movimiento ascendente y registro del tiempo del sistema.
- Registro del tiempo del sistema al final del movimiento, cálculo de variación temporal y envío del dato por terminal serie.
- Incremento de la tabla hacia nuevos parámetros de aceleración, velocidad y desplazamiento.
- Repetición del ciclo.

Un ordenador conectado al equipo ejecuta un script de Python que guarda los datos recibidos en un archivo.

En la figura 4.6 se observa una comparación de tiempos teóricos respecto a tiempos registrados en el sistema. En el eje Y se representa el tiempo en milisegundos necesario para ejecutar cada movimiento y en el eje X la velocidad. Los pares de puntos cercanos representan el movimiento descendente y ascendente respectivamente, con el mismo parámetro de velocidad y aceleración. El gráfico compara los tiempos teóricos respecto a los tiempos registrados. A simple vista no se pueden visualizar diferencias significativas.

Se presenta en la figura 4.7 un gráfico que representa los errores relativos porcentuales de las mediciones realizadas. Se puede observar que existe un aumento del error relativo a velocidades altas, con un registro pico en la velocidad de 800 mm/min.

Se concluye con este ensayo que el equipo es muy preciso en la mayor parte del rango para el cual fue diseñado teniendo un error relativo pico de 13% en las velocidades superiores del rango de funcionamiento.

4.2.2. Ejecución de comandos

El objetivo del siguiente ensayo fue demostrar que el único comando que se procesa con el equipo en funcionamiento es el de stop.

TABLA 4.1. Ensayo de tiempos en desplazamiento

Velocidad (mm/min)	Aceleración-Desaceleración(m/min)	Desplazamiento(mm)
1 mm/min	100-500-1000-2100 m/min ²	50 mm
10 mm/min	100-500-1000-2100 m/min ²	50 mm
100 mm/min	100-500-1000-2100 m/min ²	50 mm
200 mm/min	100-500-1000-2100 m/min ²	50 mm
500 mm/min	100-500-1000-2100 m/min ²	50 mm
800 mm/min	100-500-1000-2100 m/min ²	50 mm

- Configuración de movimiento descendente con valores de velocidad, aceleración y desplazamiento.
- Ejecución del movimiento descendente y registro del tiempo del sistema.
- Registro del tiempo del sistema al final del movimiento, cálculo de variación temporal y envío del dato por terminal serie.
- Configuración y ejecución de movimiento ascendente con valores de velocidad, aceleración y desplazamiento.
- Ejecución del movimiento ascendente y registro del tiempo del sistema.
- Registro del tiempo del sistema al final del movimiento, cálculo de variación temporal y envío del dato por terminal serie.
- Incremento de la tabla hacia nuevos parámetros de aceleración, velocidad y desplazamiento.
- Repetición del ciclo.

Un ordenador conectado al equipo ejecuta un script de Python que guarda los datos recibidos en un archivo.

En la figura 4.6 se observa una comparación de tiempos teóricos respecto a tiempos registrados en el sistema. En el eje Y se representa el tiempo en milisegundos necesario para ejecutar cada movimiento y en el eje X la velocidad. Los puntos cercanos representan el movimiento descendente y ascendente con el mismo parámetro de velocidad y aceleración, por tal motivo se visualizan pares de puntos cercanos. El gráfico compara los tiempos teóricos respecto a los tiempos registrados. A simple vista no se pueden visualizar diferencias significativas.

Se presenta en la figura 4.7 un gráfico que representa los errores relativos porcentuales de las mediciones realizadas. Se puede observar que existe un aumento del error relativo a velocidades altas, con un registro pico en la velocidad de 800 mm/min.

Se concluye con este ensayo que el equipo es muy preciso en la mayor parte del rango para el cual fue diseñado teniendo un error relativo pico de 13% en las velocidades superiores del rango de funcionamiento.

4.2.2. Ejecución de comandos

El ensayo se realizó para verificar que solamente el comando stop es procesado cuando el equipo está en funcionamiento.

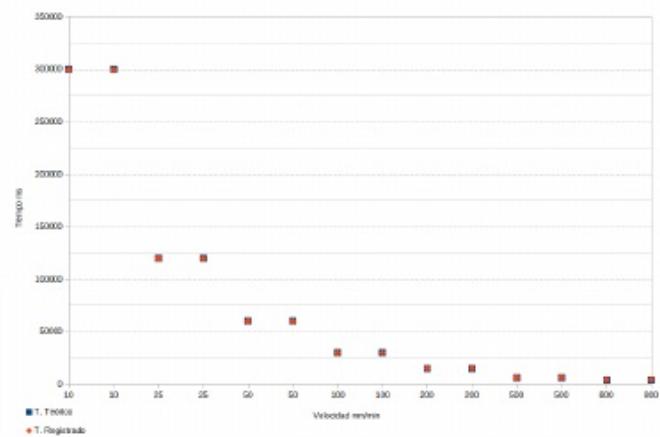


FIGURA 4.6. Comparación de tiempos teóricos y registrados.

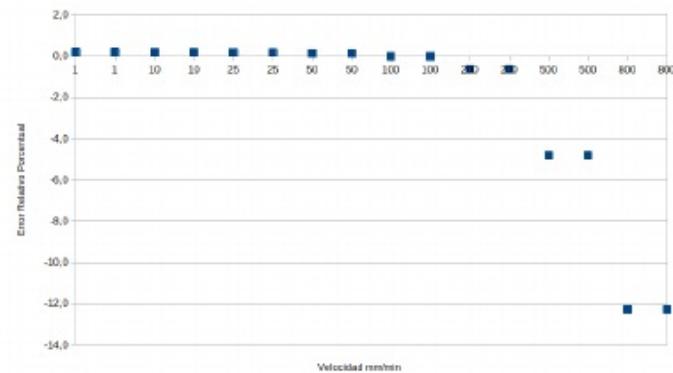


FIGURA 4.7. Error relativo porcentual.

Cada vez que el usuario ejecuta un comando sobre la consola, la aplicación app console procesa y envía el mensaje a través de una [cola](#) de FreeRTOS hacia la aplicación app coating. Si la máquina está ejecutando un movimiento individual o un proceso dip coating completo y recibe un comando nuevo, el mismo por seguridad es descartado, es decir que los mensajes no se encolan.

Para realizar el ensayo se implementó el siguiente procedimiento:

1. Ejecución de proceso dip coating.
2. Envío de comando DOWN. [Cualquier otro comando de la sección 3.2.2 tendrá el mismo efecto.](#)

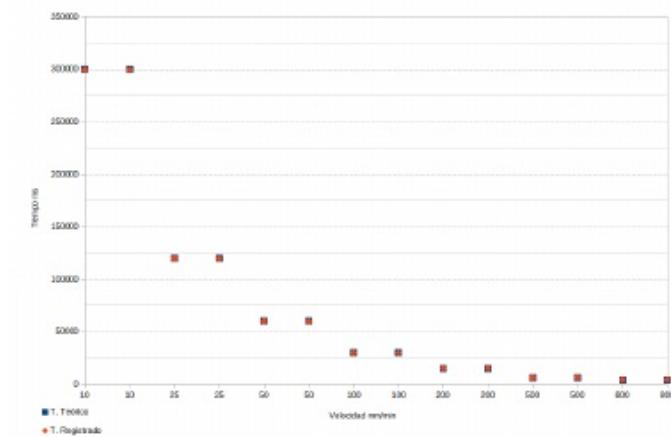


FIGURA 4.6. Comparación de tiempos teóricos y registrados.

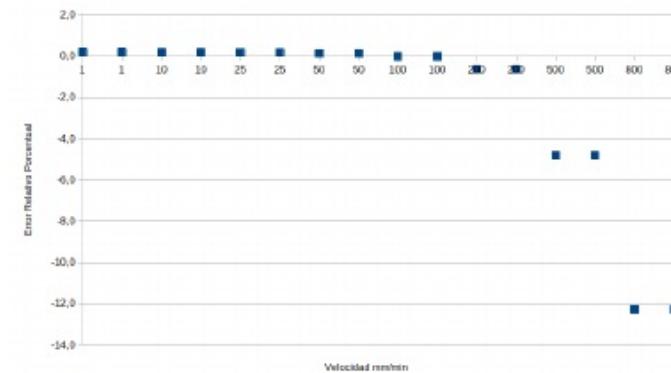


FIGURA 4.7. Error relativo porcentual.

Cada vez que el usuario ejecuta un comando sobre la consola, la aplicación app console procesa y envía el mensaje a través de una [queue](#) de FreeRTOS hacia la aplicación app coating. Si la máquina está ejecutando un movimiento individual o un proceso dip coating completo y recibe un comando nuevo, el mismo por seguridad es descartado, es decir que los mensajes no se encolan.

Para realizar el ensayo se implementó el siguiente procedimiento:

1. Ejecución de proceso dip coating.
2. Envío de comando DOWN.