eman ta zabal zazu
Universidad del País Vasco
Euskal Herriko Unibertsitatea
BILBOKO INGENIARITZA ESKOLA
ESCUELA DE INGENIERÍA DE BILBAO

GRADO EN INGENERÍA EN TECNOLOGÍA DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

DISEÑO DE UNA PLACA DE SINCRONIZACIÓN PARA SISTEMAS SoPC EN INDUSTRIA 4.0

Alumno: Castillero Martínez, Joseba

Director: Lazaro Arrotegui, Jesús

Curso: 2018-2019

Fecha: 17 de julio de 2019

RESUMEN

Este proyecto desarrolla un módulo de expansión para la plataforma *RELY-CPPS* de *RELYUM* que integre el hardware necesario para convertir al sistema en un reloj maestro. Para ello es necesario que el dispositivo *RELY-CPPS* sea capaz de integrar un servidor *NTP/PTP* que permita sincronizar un número de clientes esclavos conectados a la red. Mientras que el servidor se implementa en la propia plataforma, todo el hardware necesario para dar soporte a la aplicación le corresponde a la unidad de expansión.

Por tanto, el módulo se basa en un receptor de *GPS* que se conecta a la red de satélites de la cual obtiene la fecha y hora *UTC* y la señal *PPS* de manera muy precisa. Con ambas se puede sincronizar un sistema externo y son esenciales para crear el servidor *NTP/PTP*.

A lo largo de este documento se detalla todo el proceso de desarrollo que lleva al diseño de una tarjeta de circuito impreso que proporcione el hardware necesario para llevar a cabo la aplicación de sincronización.

PALABRAS CLAVE

Reloj maestro, sincronización, *Network Time Protocol, Precision Time Protocol*, Sistema de Posicionamiento Global, Sistema Global de Navegación por Satélite, *RELY-CPPS*, conmutador industrial, receptor de *GPS*, Placa de Circuito Impreso, *NMEA*, Pulso Por Segundo, *daemon GPSd, Raspberry Pi 3*, *Trimble ICM STM 360*.

LABURPENA

Proiektu honek RELYUM-aren RELY-CPPS plataformarako hedapen-modulua garatzen du. Moduluak lehen aipatutako sistema maisu-erlojua bihurtzeko beharrezko hardwarea integratuko du. Horretarako, RELY-CPPS gailua NTP/PTP zerbitzari bat integratzeko gai izatea, sarean konektatutako bezero esklabo kopuru bat sinkronizatzera baimenduz, beharrezkoa izango da. Zerbitzaria bereko plataforman inplementatzen den ahala aplikazioari laguntza edo sostengu ematen den beharreko hardwarea hedapen-unitatera dagokio.

Beraz, modulua satelite sarean konektatzen den GPS hartzaile batean oinarritzen da eta sare horretatik UTC data eta ordua eta PPS seinalea oso era zehatzean lortu ditzake. PPS eta UTC seinale biekin kanpo-sistema sinkroniza daiteke eta, gainera, NTP/PTP zerbitzaria sortzeko funtsezkoak dira.

Dokumentu honetan inpreso-zirkuitu txartel baten diseinura eramaten duen garapen prozesu guztia eta sinkronizazio-aplikazioa burutzeko beharko den hardwarea zehazten da.

HITZ-GAKOAK

Erloju nagusia, sinkronizazioa, *Network Time Protocol, Precision Time Protocol*, Posizionatze Globaleko Sistema, *Global Navigation Satellite System*, *RELY-CPPS*, kommutagailu industriala, *GPS*-hargailua, Inpreso-zirkuito Plaka, *NMEA*, *Pulse Per Second*, *daemon GPSd, Raspberry Pi 3*, *Trimble ICM STM 360*.

ABSTRACT

This project develops an expansion module for the RELY-CPPS platform which implements the hardware needed to transform it into a master clock. In order to become a master clock, the RELY-CPPS device must have an NTP/PTP server that allows several slave clients in the network to be synchronized. While the server is developed in the platform's system, the expansion unit must take care of all the hardware that supports the application.

Therefore, the hardware is based on a *GPS* receiver connected to a satellite network from which obtains a highly accurate *UTC* date and time and *PPS* signal. An external system can be synchronized using both and they are essential for the development on the *NTP/PTP* server.

Throughout this document, the entire process that leads to the design of a Printed Circuit Board whose purpose is to provide the essential hardware to carry out the synchronization application is detailed.

KEYWORDS

Master clock, synchronization, Network Time Protocol, Precision Time Protocol, Global Positioning System, Global Navigation Satellite System, *RELY-CPPS*, industrial switch, *GPS* receiver, Printed Circuit Board, *NMEA*, Pulse Per Second, *daemon GPSd, Raspberry Pi 3, Trimble ICM STM 360*.

LISTA DE ACRÓNIMOS

Se listan a continuación los acrónimos que se utilizan en el documento en orden de aparición:

- *CPPS*: *Cyber-Physical Production System Platform*.
- *NTP: Network Time Protocol*.
- *GPS: Global Positioning System.*
- *GNSS: Global Navigation Satellite System.*
- *UTC: Coordinated Universal Time.*
- *PTP: Precision Time Protocol.*
- *SoC: System on Chip.*
- *SMD: Surface Mount Device.*
- *SMT: Surface Mount Technology.*
- *TSIP: Trimble Standard Interface Protocol.*
- *NMEA: National Marine Electronics Association.*
- *PCB: Printed Circuit Board.*
- *BOM: Bill of Materials.*
- *CNC: Computer Numeric Control.*

ÍNDICE DE CONTENIDO

ÍNDICE DE CAPTURAS

ÍNDICE DE TABLAS

1 MEMORIA

1.1 INTRODUCCIÓN

El mantenimiento de una señal de reloj de referencia es un aspecto crítico de los sistemas conectados a internet hoy en día. La falta de sincronización es responsable de fenómenos que generan errores importantes en los dispositivos y tener una marca horaria es la clave para el análisis, la determinación de cuando han ocurrido los errores y para encontrar correlaciones entre los errores y los sucesos que los han provocado. Si un dispositivo conectado a una red esta desincronizado apenas unos milisegundos puede ocasionar que la sucesión de los eventos que ha dado lugar a un error sea muy difícil de determinar y, por tanto, muy improbable llegar a la corrección del problema.

La seguridad de las redes es otro factor importante para tener en cuenta, especialmente hoy en día que es cada vez más común tener muchos dispositivos interconectados. Los registros de tiempo pueden ayudar a diagnosticar ataques o descubrir vulnerabilidades que pueden ser explotadas. Sin embargo, un registro poco preciso, o la falta de uno, convierte el proceso de diagnóstico en una tarea inabarcable.

Por tanto, en este proyecto se pretende desarrollar una placa que pueda ser implementada como uno de los módulos de la plataforma *RELY-CPPS* de *RELYUM,* que proporcione las funcionalidades de sincronización propias de un reloj maestro.

1.2 CONTEXTO

La plataforma *RELY-CPPS* (*Cyber-Physical Production System Platform*) es un sistema que integra un conmutador industrial de Ethernet que además combina la adquisición de datos de sensores, la capacidad de guardar y procesar los datos en el propio dispositivo mejorando el tiempo de respuesta (*Edge Computing*).

Captura 1. Captura de la apariencia del sistema RELY-CPPS montado.

Además, dispone de módulos de expansión opcionales que añaden funcionalidades que cubren la mayoría de las aplicaciones propias de los sectores de energía e industria como puede ser un módulo que añade nuevas entradas y salidas de diferentes niveles lógicos usados en la industria para sensores o un módulo que facilita conexiones inalámbricas de estándares de *WIFI 802.11* y *Bluetooth v4.0*. El objetivo de disponer de esta capacidad de personalización que ofrece la modularidad de la plataforma es ofrecer el dispositivo más apropiado para cada cliente.

1.3 OBJETIVOS, REQUERIMIENTOS Y ALCANCE DEL TRABAJO

El objetivo de este proyecto es dotar a la plataforma *RELY-CPPS* de un módulo que sea capaz de proporcionar las funcionalidades propias de un sistema que tenga acceso una señal de reloj maestro. Como el sistema *RELY-CPPS* está dotado de funcionalidades de red, la sincronización para la mayoría de las aplicaciones cobra mucha importancia. Para ello, se basa en el sistema para la sincronización de relojes usando protocolos estandarizados como el *Network Time Protocol* (*NTP*) proporcionado por cualquier Sistema de Posicionamiento Global (*GPS*) o Sistema Global de Navegación por Satélite (*GNSS*). Para integrar estas capacidades se decide usar un receptor *GPS* que proporcione el torrente de información que incluye datos como posición, hora, fecha, zona horaria, etc.

La razón por la cual es necesario utilizar un hardware para sincronizar el sistema y crear el servidor es que al utilizar dispositivos hardware específicos la red minimiza la latencia, siendo crucial en un sistema sincronizado. Las soluciones que utilicen software para obtener datos de sincronización a través de internet, por ejemplo, se enfrentarán a factores que dificultan mantener unos registros precisos como problemas en las redes de comunicación o que gasten recursos de los sistemas operativos de forma rutinaria para funciona. Mientras que un sistema conectado a la red de satélites obtiene la capacidad de comunicarse con sistemas que tienen un reloj atómico de la mejor calidad que constantemente retransmite la hora y la posición del satélite, de forma que al estar los satélites sincronizados entre sí un receptor puede obtener información de varios satélites para triangular su posición, obtener la hora y zona horaria en la que se encuentra. Todas estas características convierten al receptor de GPS en un sistemas mucho más seguro y preciso que cualquier aplicación software de sincronización existente.

El protocolo *NTP* proporciona una señal de reloj de referencia junto con el Tiempo Universal Coordinado (*UTC*), de manera que con un módulo receptor de *GPS* o *GNSS* se puede convertir al módulo del sistema *RELY-CPPS* en un servidor *NTP* que pueda aceptar cualquier número de dispositivos clientes. Un servidor *NTP* permite a los clientes que sincronicen sus relojes y otros dispositivos de temporización con gran precisión sin la necesidad de estar conectados a internet. Por lo que colocando un servidor *NTP* en una red local protegida por un firewall aumenta la precisión, redundancia, confiabilidad y seguridad.

Para la mayoría de las aplicaciones un servidor *NTP* que proporciona sincronización a nivel del milisegundo es más que suficiente. No obstante, la plataforma *RELY-CPPS* está preparada para hacer frente a las aplicaciones más exigentes por lo que integra la capacidad de migrar a un servidor que utilice el *Precision Time Protocol (PTP)*. El *PTP* es otro protocolo de sincronización

estandarizado para redes del estilo maestro-esclavo. Las redes *PTP* consiguen precisiones superiores a aquellas basadas en *NTP*, alcanzando niveles de nanosegundo o incluso picosegundo. El motivo por el que esta sincronización es tan precisa es porque los equipos *PTP* tienen un único objetivo: mantener los dispositivos sincronizados, incluso tienen en cuenta la latencia de la red.

Por tanto, es importante que el módulo de sincronización que se va a diseñar en este proyecto integre la capacidad de funcionar tanto como un servidor *NTP* como un servidor *PTP*. Sin embargo, este proyecto no pretende cubrir la creación de un servidor de sincronización. Para terminar el desarrollo del módulo de sincronización basta con elegir una plataforma de desarrollo accesible para sincronizarla con el receptor de *GPS* demostrando el funcionamiento del sistema.

1.4 BENEFICIOS QUE APORTA EL TRABAJO

Como se ha mencionado previamente, incorporar un servidor *NTP/PTP* que maneje la sincronización de una red tiene grandes beneficios:

- Creación y mantenimiento de registros de tiempo que facilitan el análisis de problemas que puedan surgir en la red.
- Mayor facilidad para determinar la secuencia de cualquier serie de eventos.
- Mejor seguridad en la red.
- Mayor nivel de precisión que cualquier reloj interno o servidor público.

La mejor forma de asegurar que la sincronización de una red sea precisa, consistente y segura es utilizar un servidor *NTP/PTP* privado propio. Esta solución no necesita reconfigurar *routers* o *firewalls* para permitir que los datos del servidor público lleguen sin problemas a la red local.

Por tanto, es la mejor solución si se quiere mejorar la sincronización en cualquier aplicación que utilice la plataforma *RELY-CPPS*.

1.5 ANÁLISIS DE ALTERNATIVAS

La piedra angular del diseño es sin duda el módulo receptor *GNSS* en el que se basa todo el sistema. Existen bastantes alternativas que ofrecen las funcionalidades propias del sistema integrado en un único chip (del inglés: *System on Chip*, *SoC*). Un *SoC* no es solo más pequeño, sino que es más seguro, con mejores funcionalidades, menor consumo y mayor facilidad de implementación. Por tanto, es necesario realizar un estudio del mercado (ver Anexo I) previo para analizar las diferentes ofertas y seleccionar el más indicado.

Tras la evaluación de las alternativas se determina que el módulo receptor multi-GNSS de *Trimble ICM SMT 360* es el que mejor se adapta a las necesidades planteadas por su pequeño tamaño, el formato de montaje superficial (en inglés: *Surface-Mount Device, SMD*), su diseño para funcionar en áreas urbanas muy pobladas con una visión limitada del cielo y los niveles lógicos de las señales compatibles con la plataforma *RELY-CPPS*. Debido a que se desea desarrollar una

tarjeta que debe tener un tamaño igual que el resto de los módulos del sistema ya disponibles, la cualidad más destacable es el reducido tamaño que posee el *SoC*, ya que el resto de las alternativas son prohibitivas en este aspecto.

La única desventaja que presenta elegir esta alternativa es el protocolo de comunicaciones a través del cual se trasmiten los datos de la red *GNSS*, *Trimble Standard Interface Protocol* (*TSIP*). Por tanto, si se requiere interactuar con el módulo, por ejemplo, para cambiar su configuración este solo acepta mensajes con la estructura propia del protocolo *TSIP*. Sin embargo, existe la forma de establecer que los mensajes enviados por el módulo tengan formato *NMEA* con un simple mensaje de configuración. Este estándar de comunicaciones propiedad de *National Marine Electronics Association* (*NMEA*) ha sido desarrollado para la transmisión de datos de distintos dispositivos electrónicos navales como sonares, anemómetros y receptores de *GPS* entre otros. Por tanto, se soluciona el problema del protocolo específico del chip.

Además, está disponible integrado en una placa base con un conector de Radio Frecuencia (*RF*) para la antena. El sistema está diseñado para conectarse automáticamente a una red *GNSS* y proporcionar las distintas señales a través del conector 2x4. Aún montado en la placa sigue siendo la solución más compacta entre las opciones estudiadas y su tamaño es inferior al tamaño máximo de debe tener el módulo.

Captura 2. Apariencia de la parte superior del módulo ICM STM 360 integrado.

Captura 3. Apariencia de la parte inferior del módulo ICM STM 360 integrado.

Por lo que, utilizando esta implementación es fácil crear un prototipo sencillo y barato que simplemente alimente la placa y de acceso a las señales. Por tanto, lo único que hay que diseñar es una adaptación a la topología propia de los módulos del sistema *RELY-CPPS* junto con una protección básica de la alimentación, ya que la tarjeta que integra el módulo ICM STM 360 tiene incluidos circuitos de protección y de detección del estado de la antena.

1.6 DISEÑO BÁSICO

Si la tarjeta que integra el módulo *ICM STM 360* proporciona todas las funcionalidades que se necesitan, el diseño simplemente debe adaptar del conector de una placa a otra. Para ello, se diseña una placa de circuito impreso (del inglés: *Printed Circuit Board*, *PCB*) que conecte la tarjeta base del sistema *RELY-CPPS* con el *PCB* que tiene el módulo *ICM STM 360*. Por tanto, la parte más fundamental de los componentes que tiene esta tarjeta son los conectores, ya que tienen que ser compatibles. Si los conectores no son compatibles, o bien no están correctamente situados, la tarjeta que se produzca no se

El sistema para el que debemos diseñar el módulo de reloj maestro es un dispositivo pensado para poder incorporar hasta cuatro tarjetas distintas. Por tanto, existen tarjetas ya desarrolladas de las cuales se puede basar el diseño de este proyecto, por ejemplo, la placa *S-3319B*. Se pueden obtener los conectores de las tarjetas modulares compatibles con la plataforma *RELY-CPPS*, así como las protecciones de la alimentación que incluyen todos los módulos del diseño de la tarjeta.

Por otro lado, el manual de usuario del *SoC ICM STM 360* proporciona el conector de 8 pines que utiliza, fabricado por *Samtec* y de código *TMM104-01-T-D-SM*. Este conector tiene las señales que se muestran en la siguiente tabla:

Pin	Función	Descripción
1	Alimentación antena	3.0 – 5.5 VDC, 55 mA máximo
2	Alimentación principal	3.3 ± 0.3 VDC, 110 mA
3	TXDA	Transmisión serie, LVTTL
4	SYSCLK	Señal de salida de 10 MHz, LVTTL
5	RXDA	Recepción serie, LVTTL
6	PPS	Pulse Per Second, LVTTL
7	Reservado	
8	GND	Tierra

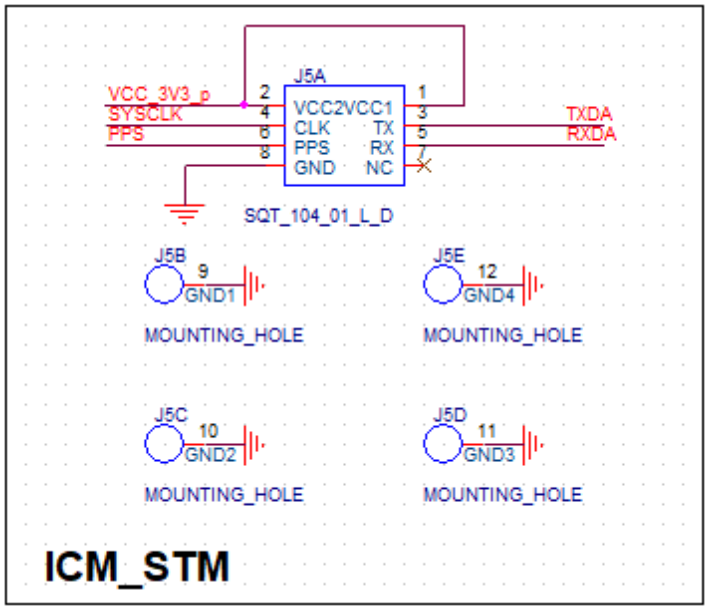
Tabla 1. Señales de entrada/salida del ICM STM 360 montado en PCB.

A pesar de tener una alimentación principal independiente de la alimentación propia de la antena, tienen rangos de tensión compatible y el manual del usuario especifica que no es necesario seguir una secuencia al encender o apagar el dispositivo. Por lo que pueden juntarse en una única alimentación de 3.0 – 3.6 V_{DC} y 165 mA de consumo. En cuanto al resto de señales, basta

con conectarlas a los pines de los cuatro conectores que se conectaran con la tarjeta base del sistema *RELY-CPPS*.

Teniendo en cuenta todos los requisitos mencionados hasta ahora, se puede pasar al diseño inicial del esquemático (ver Anexo II) necesario para el diseño posterior del *PCB*. Se usa para su diseño una licencia pública proporcionada por la universidad del programa de desarrollo *OrCAD Capture*.

La parte más importante del esquemático es el conector de la tarjeta *ICM STM 360*, ya que es la parte que no se hereda de otras tarjetas. Por tanto, es fundamental elegir un conector compatible con el *TMM104-01-T-D-SM* que lleva montado el *PCB*. Siendo el fabricante *Samtec* se puede encontrar el conector recomendado por el propio fabricante: *SQT-104-01-L-D*. También hay que tener en cuenta los cuatro agujeros de los bordes de la tarjeta que no solo sirven como soporte para la conexión, sino que también están conectados a tierra, por lo que hay que representarlos en el esquemático. El equivalente del componente que se obtiene en el diseño esquemático se genera como una única pieza compuesta por 5 partes, el conector de 8 pines compatible junto con los 4 agujeros para los soportes.



Captura 4. Representación en el diseño esquemático de la tarjeta ICM STM 360.

Además, es necesario seleccionar los componentes que son parte del esquemático. Incluso los que ya están decididos, como los 4 conectores principales, hay que buscarlos en un distribuidor para su posterior adquisición. Es importante en este punto del desarrollo del proyecto tener la lista completa de todos los materiales para referenciar cada componente del esquemático con su componente real. Esa lista denominada *Bill of Materials* (*BOM*) puede ser creada por el propio programa de desarrollo que se ha empleado hasta ahora.

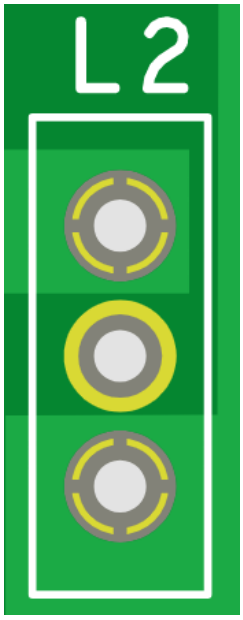
El *BOM* (ver Anexo III) que genera tiene todos los componentes que forman parte del esquemático y sus valores, por tanto, solo hay que rellenar la referencia del fabricante y la referencia del

distribuidor. En este proyecto se elige a *Farnell* como distribuidor y a través de su buscador se seleccionan todos los componentes salvo el *ICM STM 360*. Casi la totalidad de los componentes seleccionados son de montaje superficial salvo en los casos donde no existe alternativa y hay que recurrir a partes de agujero pasante (ver Anexo IV). Como en este proyecto se está diseñando un prototipo y la producción no es de cientos de tarjetas, es fundamental priorizar el menor precio y la posibilidad de pedir pequeñas cantidades de cada dispositivo.

Cabe destacar también que el *BOM* es un documento importante, ya que, si se opta por delegar el montaje a una empresa que se dedique a ello, es uno de los archivos que necesitan para comprar los componentes y relacionar cada pieza con su lugar de montaje en la tarjeta.

Con todos los componentes seleccionados hay que crear su huella, que dependerá del encapsulado en el que viene el dispositivo en cuestión. Es un paso importante ya que delimita la zona en la que se va a montar en el *PCB* y de existir un fallo de dimensiones dificulta en exceso la soldadura o la hace imposible, convirtiendo el prototipo en una placa inútil que hay que rehacer. Existen huellas que no necesitan ser creadas, ya que pertenecen a estándares de la industria con dimensiones establecidas y usadas ampliamente. Por ejemplo, las resistencias y condensadores usan el formato *0603* y los componentes *D26* y *D27* el encapsulado *SOT-23*. Estas huellas existen en las librerías básicas de cualquier programa para generar huellas. Sin embargo, algunas partes como los conectores o los agujeros para los separadores no existen y hay que generarlos desde cero.

El proceso para obtener una huella consta de dos partes distintas. Primero hay que generar el *pad*, la superficie de cobre expuesta que permite soldar el componente al *PCB*, y la máscara de soldadura, la capa de polímero que rodea los *pads* que protege contra la oxidación y previene que se creen puentes entre *pads*. Las dimensiones de los *pads* de cada componente se obtienen de la ficha técnica de estos, donde también se especifican la disposición de estos para formar el conjunto de la huella.



Captura 5. Captura de la previsualización de los pads del componente L2.

Para crear el *pad* y la máscara de soldadura se emplea el programa *Padstack Editor* de *OrCAD* que permite introducir las dimensiones, la forma del *pad* y el grosor de la máscara de soldadura que lo rodea. Mientras que algunas dimensiones como el diámetro del agujero o la anchura mínima del *pad* son dadas por el fabricante del dispositivo, existen otros requisitos impuestos por el fabricante del *PCB* que deben cumplirse.

Dependiendo de la precisión y la tolerancia necesarias para fabricar la placa final el precio puede variar. Obviamente, cuanto más pequeña sea la máscara de soldadura, por ejemplo, más precisión y menor tolerancia requieren, aumentando el coste por *PCB*. Teniendo en cuenta que el diseño de la tarjeta de este proyecto es bastante simple y no requiere de mucha precisión, se opta por acogerse a la categoría de menor precisión y tolerancia para abaratar lo máximo posible la producción. Se elige *Eurocircuits* para la fabricación del *PCB*, por lo que hay que atenerse a la clase más barata posible que ofrecen.

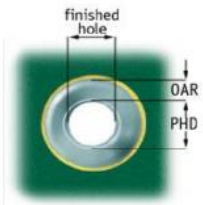
Los valores mínimos de los taladros de la menor clase, la clase A, son más pequeños que los usados por los componentes seleccionados, por lo que no hay ninguna restricción que impida usar esa categoría. Solo hay que tener en cuenta la anchura mínima del anillo de cobre alrededor de un agujero y la anchura mínima de la máscara de soldadura de la clase 3 al crear los *pads* para estos componentes. Las tablas con los valores para las distintas clases se pueden encontrar en la página web de *Eurocircuits* como se muestran a continuación.

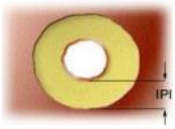
Eurocircuits - PCB design classification overview

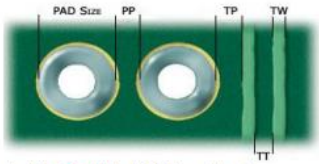
Pattern Class	class 3		class 4		class 5		class 6		class 7		class 8		class 9		class 10		
Service	P+S+R+I		P+S+R+I		P+S+R+I		P+S+R+I		S+R		S+R		S+R		-		
OTW	0.250	10	0.200	8	0.175	7	0.150	6	0.125	5	0.100	4	0.090	3.5	<0.090	<3.5	mm-mil
OTT-OTP-OPP	0.250	10	0.200	8	0.175	7	0.150	6	0.125	5	0.100	4	0.090	3.5	<0.090	<3.5	mm-mil
OAR	0.200	8	0.150	6	0.150	6	0.125	5	0.125	5	0.100	4	0.100	4	<0.100	<4	mm-mil
ITW	0.250	10	0.200	8	0.175	7	0.150	6	0.125	5	0.100	4	0.090	3.5	<0.090	<3.5	mm-mil
ITT-ITP-IPP	0.250	10	0.200	8	0.175	7	0.150	6	0.125	5	0.100	4	0.090	3.5	<0.090	<3.5	mm-mil
IAR	0.200	8	0.150	6	0.150	6	0.125	5	0.125	5	0.125	5	0.125	5	<0.125	<5	mm-mil
IPI	0.275	11	0.225	9	0.225	9	0.200	8	0.200	8	0.200	8	0.200	8	<0.200	<8	mm-mil

The smallest value (OTW, OTT-OTP-OPP, OAR, ITW, ITT-ITP-IPP, IAR, IPI) determines the **Pattern Class** of the board

Base Cu		min Pattern values				
Base Cu OL		OTT-OTP-OPP		OTW		
12µm	⅓oz	0.090	3.5	0.090	3.5	mm-mil
18µm	½oz	0.125	5	0.090	3.5	mm-mil
35µm	1oz	0.175	7	0.125	5	mm-mil
70µm	2oz	0.250	10	0.200	8	mm-mil
105µm	3oz	0.300	12	0.250	10	mm-mil
Base Cu IL		ITT-ITP-IPP		ITW		
12µm	⅓oz	0.090	3.5	0.090	3.5	mm-mil
18µm	½oz	0.100	4	0.090	3.5	mm-mil
35µm	1oz	0.125	5	0.125	5	mm-mil
70µm	2oz	0.250	10	0.200	8	mm-mil
105µm	3oz	0.300	12	0.250	10	mm-mil







Preceding letters **O** and **I** stand for Outer- and Inner layer
Example: **OTW** = **O**uter layer **T**rack **W**idth

OAR : smallest **OAR** (Outer layer Annular Ring = 1/2 (Outer layer pad diameter - PHD))
IAR : smallest **IAR** (Inner layer Annular Ring = 1/2 (Inner layer pad diameter - PHD))
IPI (Inner layer **P**ad **I**nsulation) : Clearance between edge PHD of any unconnected hole(PTH/NPTH) and any nearest copper

Smallest **PHD** : Production **H**ole **D**iameter or tool size = finished hole size + 0.10mm/4mil for **P**lated **T**hrough **H**oles
+ 0.00mm/0mil for **N**on **P**lated **T**hrough **H**oles

Drill Class	class A		class B		class C		class D		class E		class F		
Service	P+S+R+I		P+S+R		P+S+R		S+R		S+R		-		
min PHD	0.60	0.026	0.45	0.018	0.35	0.014	0.25	0.010	0.20	0.008	<0.20	<0.008	mm-inch
Corresponding finished holes sizes													
PTH	0.50	0.022	0.35	0.014	0.25	0.010	0.15	0.006	0.10	0.004	<0.10	<0.004	mm-inch
NPTH	0.60	0.026	0.45	0.018	0.35	0.014	0.25	0.010	0.20	0.008	<0.20	<0.008	mm-inch

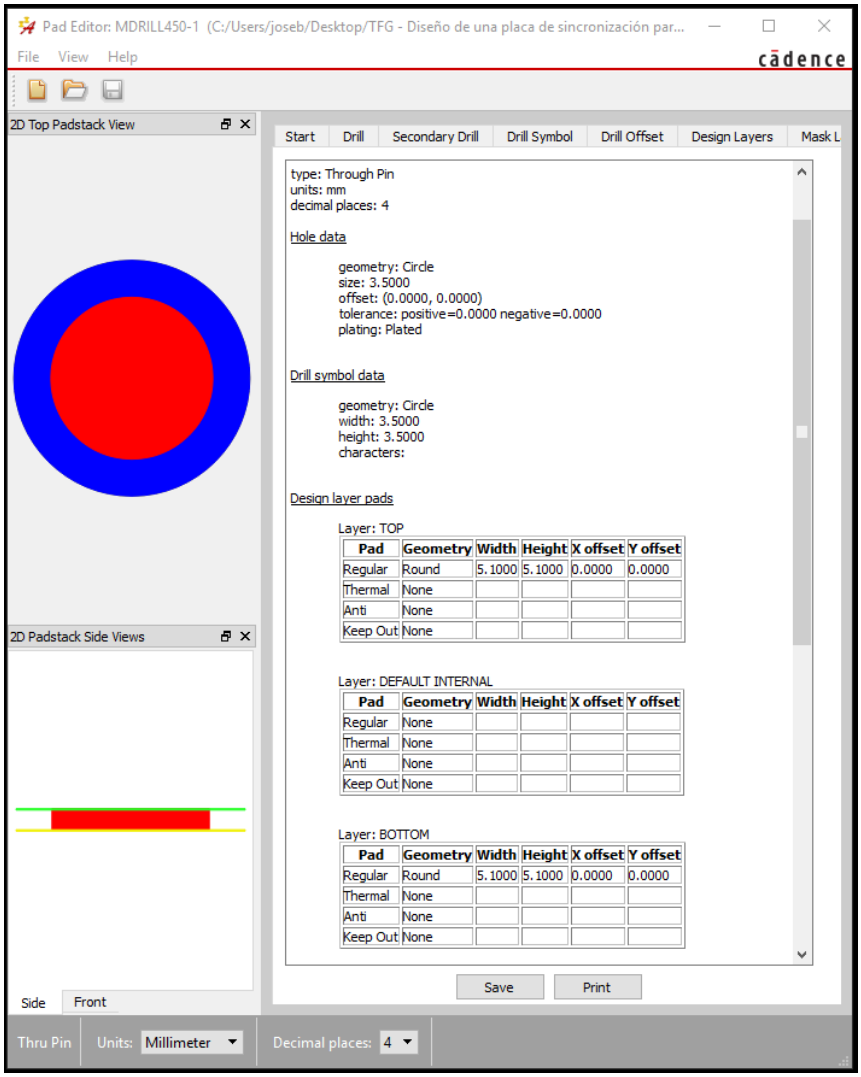
The smallest value (PHD) determines the **Drill Class** of the PCB

Max. PCB thickness to Drill Class	3.20	0.125	3.20	0.125	2.40	0.093	2.00	0.079	1.60	0.062	mm-inch	Aspect ratio is 1 / 8

Note A: VIA holes are Plated Through Holes, default defined as <=0.45mm (18mil) for all services or <= as defined by the customer in the order details.
VIA holes have a maximum negative tolerance of 0.30mm (12mil)
Note B: This classification table can only be put into praxis on PCB designs that have a **Plating Index of 0.40 or higher.** This is calculated in the PCB Visualizer analysis and displayed in the PCB Visualizer order details.
Services Index : **P** = PCB proto **S** = STANDARD pool **R** = RF pool **I** = IMS pool

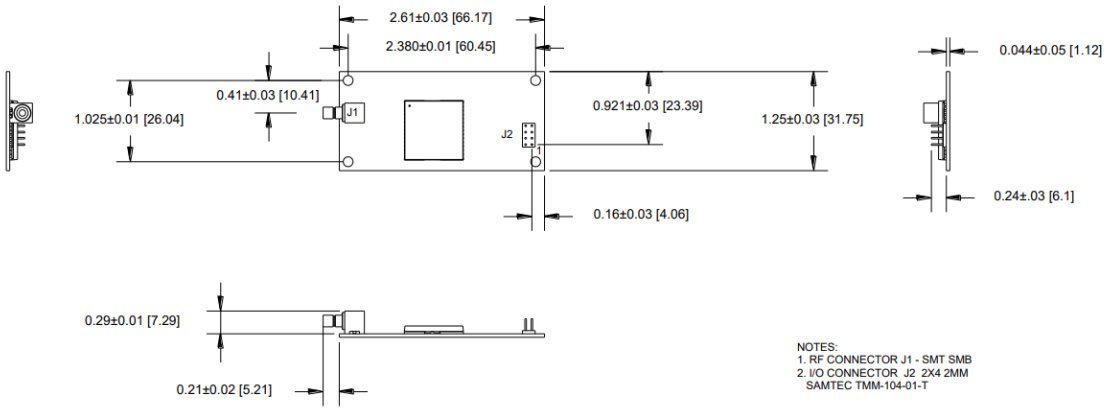
Captura 6. Clasificación del diseño de PCB de Eurocircuits.

Por tanto, si los *pads* de las huellas de los componentes no requieren ninguna restricción que evite que el prototipo pueda categorizarse como la clase más barata, se hace uso de los valores especificados en la captura anterior. Por ejemplo, la siguiente captura del programa de edición *Padstack Editor* muestra los valores de uno de los pads para los separadores de la huella del *ICM STM 360*. Se puede observar cómo se respetan los valores mínimos para pertenecer a la categoría más barata.



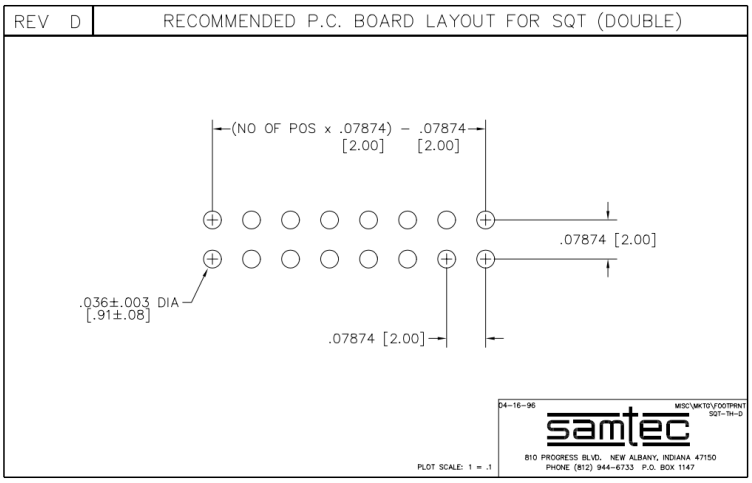
Captura 7. Diseño del pad del agujero para los soportes de la tarjeta ICM STM 360.

Una vez que se han creado todos los *pads* con sus respectivas máscaras de soldadura se puede avanzar al segundo paso, generar la huella completa del componente. El programa de desarrollo para crear una huella que se utiliza es el *PCB Editor*, también propiedad de *OrCAD*, que permite crear y editar tanto huellas de componentes como placas de circuito impreso. Por ejemplo, en el caso de la huella de la placa *ICM STM 360* se tienen que generar dos *pads* distintos: uno para los cuatro agujeros para los soportes y otro para los ocho pines del conector. La posición de los *pads* se obtiene del manual de usuario donde se especifican de las dimensiones de la tarjeta.



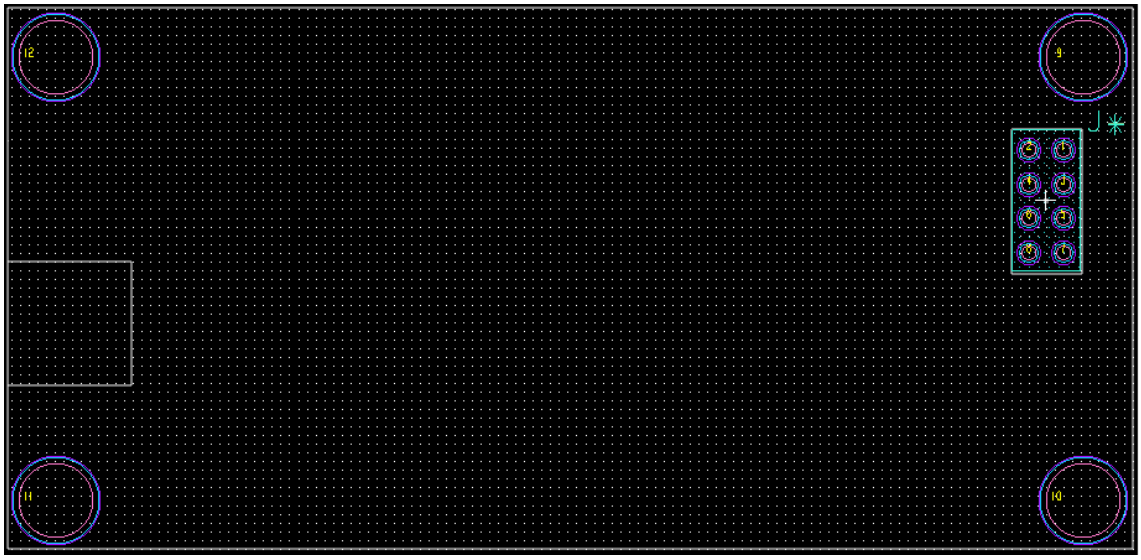
Captura 8. Captura de las dimensiones de la placa ICM STM 360 proporcionadas por el manual de usuario.

Estas medidas junto con las referentes al conector *SQT-104-01-L-D* que se obtienen de su ficha técnica permiten obtener el símbolo que representa la huella de todo el *PCB*.



Captura 9. Disposición de la huella de conectores SQT recomendada.

En este caso, se tienen que usar dos *pads* para crear la huella del componente: los *pads* más grandes para los cuatro espaciadores de las esquinas y los *pads* pequeños para los 8 pines del conector. Además de los *pads* hay que añadir la capa de serigrafía, que no es una capa esencial para el desarrollo de cualquier dispositivo, sino que es más una ayuda visual que facilita trabajar con el *PCB* y es recomendable en diseños no afianzados. Como se puede observar en la captura de la huella de la tarjeta *ICM STM 360*, la serigrafía delimita sus dimensiones.



Captura 10. Huella de la tarjeta ICM STM 360, denominada SQT-104-01-L-D.

Tras haber completado las huellas para todos los componentes del esquemático, se puede pasar al último paso antes de empezar con el diseño de la placa de circuito impreso. Los componentes del esquemático deben tener una huella asociada para poder generar la placa de circuito impreso posteriormente. Por tanto, hay que entrar en cada parte y especificar que huella le corresponde y la localización del archivo que la contiene.

	A
	⊞ SCHEMATIC1 : PAGE1
Color	Default
Designator	A
Graphic	SQT_con_5A.Normal
ID	
Implementation	
Implementation Path	
Implementation Type	<none>
Location X-Coordinate	830
Location Y-Coordinate	70
Name	INS15471287
Part Reference	J5A
PCB Footprint	SQT-104-01-L-D
Power Pins Visible	☐
Primitive	DEFAULT
Reference	J5
Source Library	C:\USERS\JOSEB\DES ...
Source Package	SQT_con_5
Source Part	SQT_con_5A.Normal
SPLIT_INST	TRUE
SWAP_INFO	(S1+S2+S3+S4+S5)
Value	SQT_104_01_L_D

Captura 11. Huella de la tarjeta ICM STM 360 referenciada a su parte en el diseño esquemático.

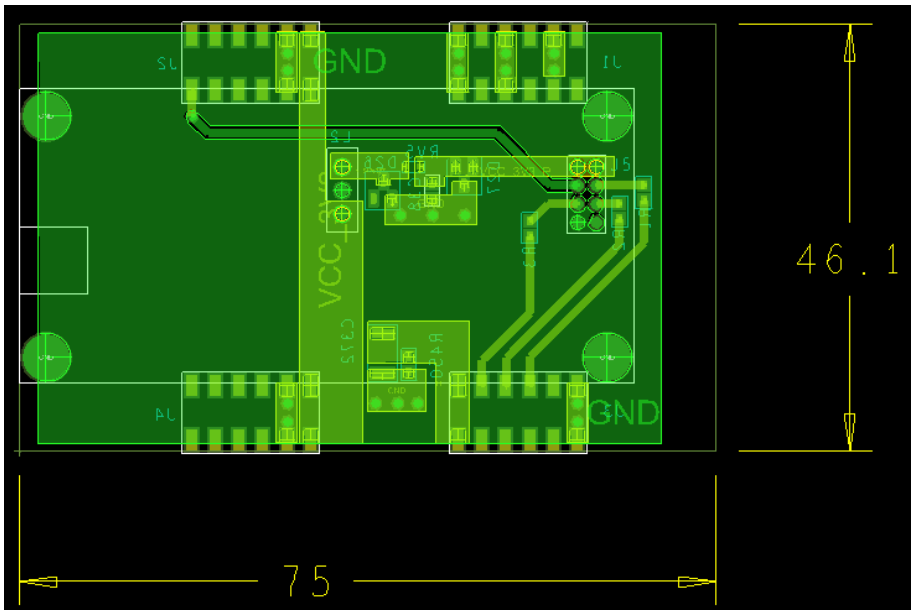
Finalmente, se utiliza la herramienta de *Capture* que permite crear un nuevo proyecto en blanco en el *PCB Editor* manteniendo los componentes con sus huellas y las conexiones electrónicas creadas en el esquemático, de forma que solo hay que distribuir correctamente los componentes por la placa y crear los caminos de cobre entre los componentes.

Como el diseño del *PCB* de este proyecto es bastante sencillo, no existen muchos componentes y, por tanto, líneas de cobre, se decide usar únicamente dos capas de cobre de 0.017 mm de grosor separadas por una capa de dieléctrico de 0.51 mm.

Objects		Types >>		Thickness	Physical >>	
#	Name	Layer	Layer Function	Value mm	Layer ID	Material
*	*	*	*	*	*	*
		Surface				
1	TOP	Conductor	Conductor	0.017	1	Copper
		Dielectric	Dielectric	0.51		Fr-4
2	BOTTOM	Conductor	Conductor	0.017	2	Copper
		Surface				

Captura 12. Capas del diseño del PCB.

Las dimensiones de la placa y la posición los conectores *J1 – J4* se obtienen del diseño de la tarjeta *S-3319B*. La disposición del módulo de GPS se puede observar en la siguiente captura, siendo necesaria colocarla de tal forma ya que, como muestra la serigrafía del componente, el conector *RF* del *ICM STM 360* debe de estar en uno de los laterales para poder conectar el cable de la antena *GNSS* desde el exterior de la caja que contiene el sistema *RELY-CPPS*.



Captura 13. Diseño final del PCB.

La reducida cantidad de líneas de cobre permite crear en solo dos capas de cobre grandes planos de potencia. Colocar un plano de tierra que ocupe casi la totalidad de la tarjeta permite mejorar el aislamiento de las señales, aumentar la conductividad y conectar los cuatro agujeros para los separadores de forma sencilla. Además, implementar el plano en la cara superior (ver Anexo V) de la tarjeta es la solución más sencilla debido a la existencia de una única línea de conexión en esa cara.

En cuanto a los planos de 3.3 V_{DC}, no se pueden extender tanto como el de tierra debido a que se dispone en la capa inferior (ver Anexo V) donde se encuentran la mayoría de las conexiones y componentes. Por tanto, se llega al compromiso de crear planos lo suficientemente grandes como para trasladar la corriente suficiente para alimentar la tarjeta *ICM STM 360*, pero sin estorbar a las líneas que llevan las señales importantes como son la comunicación serie y el *PPS*.

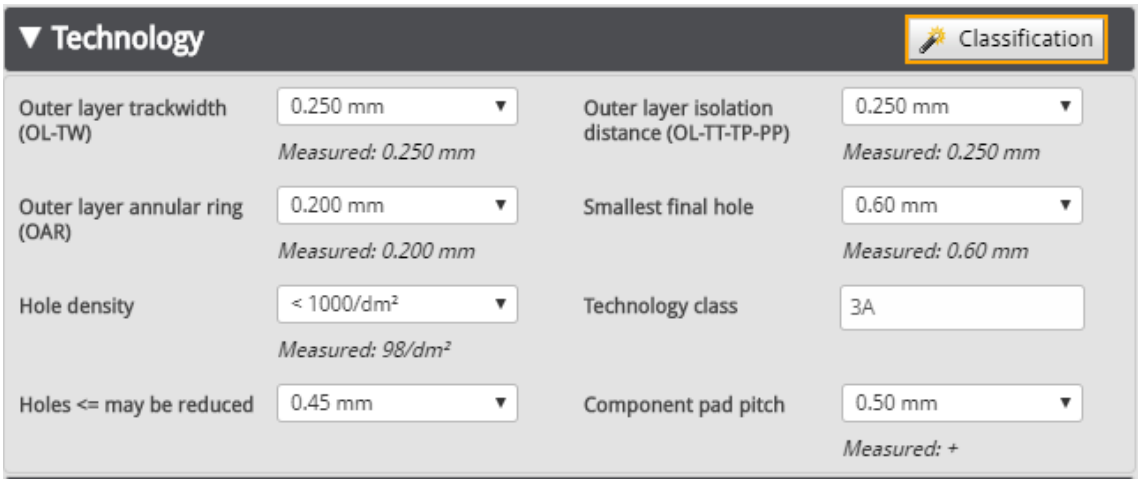
Habiendo finalizado el diseño de la placa de circuito impreso se puede pasar a la producción de prototipos. Se ha mencionado anteriormente que se elige a *Eurocircuits* como productor, por tanto, hay que proporcionarles los archivos necesarios para su fabricación. Estos archivos llamados *Gerber* tienen un formato definido como estándar para la producción de *PCBs* y son generados por el propio software de desarrollo con el que se ha diseñado la tarjeta. Se debe crear un fichero *Gerber* por cada capa existente, es decir, por la capa de cobre, la máscara de soldadura y serigrafía tanto de la parte superior e inferior (ver Anexo V), además de otro por el borde que delimita el tamaño de la placa. Únicamente es necesario un archivo extra, que guarda la posición y diámetro de los agujeros que debe taladrar en el *PCB* la máquina de control numérico (del inglés: *Computer Numerical Control, CNC*), también generado por el *PCB Editor*.

Archivo	Descripción
BOTTOM.art	Capa de cobre de la cara inferior
TOP.art	Capa de cobre de la cara superior
SMB.art	Máscara de soldadura de la cara inferior
SMT.art	Máscara de soldadura de la cara superior
SSB.art	Serigrafía de la cara inferior
SST.art	Serigrafía de la cara superior
OUTLINE.art	Contorno de la placa
TSGV0_4-1-2	CNC

Tabla 2.Lista de los archivos necesarios para la fabricación del PCB.

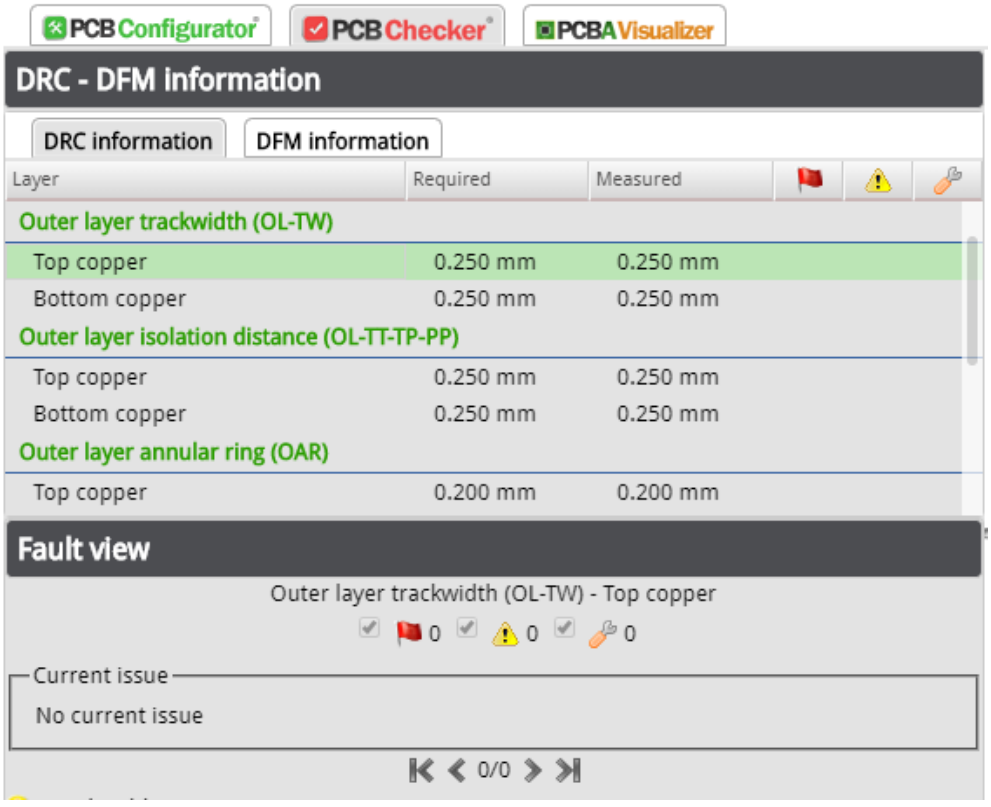
Antes de empezar con la producción es recomendable revisar los *Gerbers* con cualquier visualizador online de los que estás disponibles de manera gratuita. Uno de los motivos por el que en este proyecto se ha decidido producir el *PCB* en Eurocircuits es que proporcionan un servicio muy amplio de revisión que permite introducir los *Gerbers* y observar errores, revisar a que categoría pertenece si se desea producir y un presupuesto estimado entre otras cosas. Una vez enviados los archivos y después de que se procesen en sus servidores se puede acceder a la visualización y análisis del *PCB*.

Gracias a esta herramienta de diagnostico se comprueba que la placa desarrollada pertenece realmente a la categoria mas barata 3A.



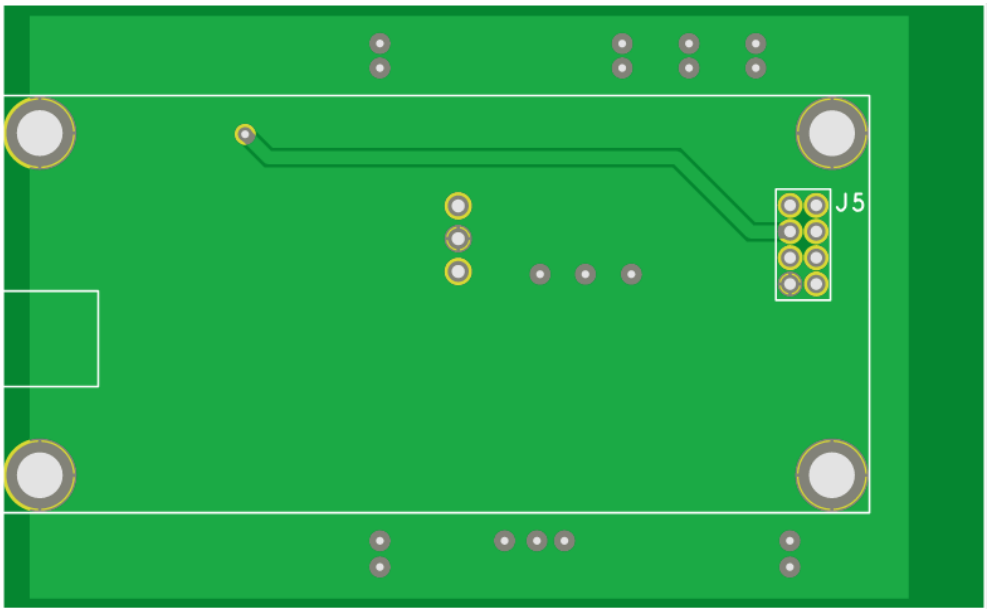
Captura 14. Captura del PCB Configurator de Eurocircuits que muestra la clase del PCB.

Se revisa si existe algún error con el *PCB Checker* que necesite de algún ajuste.

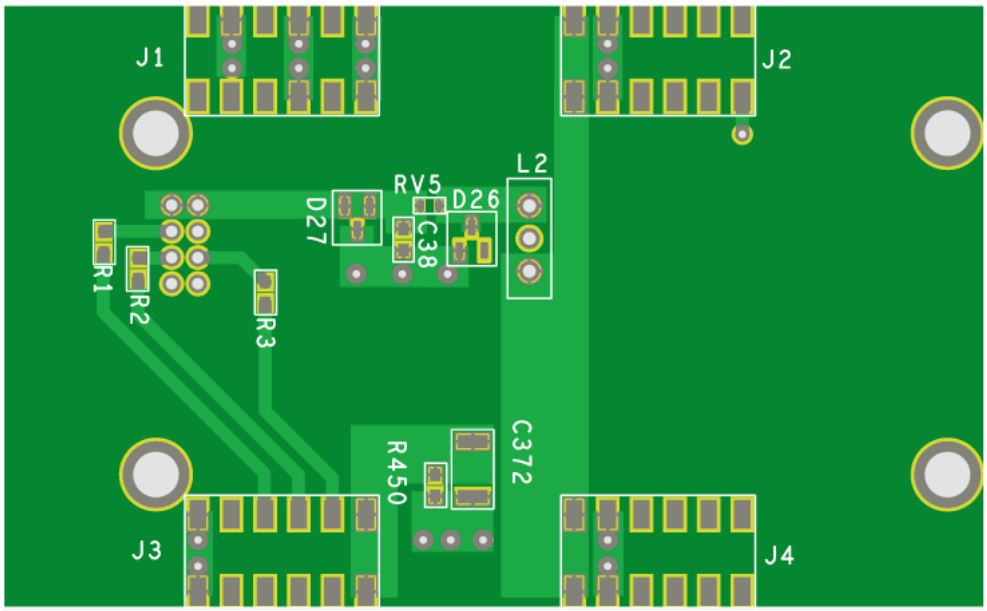


Captura 15. Revisión de errores proporcionado por el PCB Checker de Eurocircuits.

Sin embargo, esta herramienta solo es capaz de detectar errores en las medidas. Así que es necesario prestar especial atención a las huellas de los componentes para comprobar que la superposición de las distintas capas sea correcta, ya que si existe un problema que se puede apreciar en la previsualización, supone un error en el diseño. Tras revisar las previsualizaciones se concluye que no existe ningún error y se puede pasar a la producción.



Captura 16. Previsualización de la cara superior del PCB.



Captura 17. Previsualización de la cara inferior del PCB.

2 METODOLOGÍA SEGUIDA EN EL DESARROLLO DEL TRABAJO

2.1 DESCRIPCIÓN DE TAREAS, FASES, EQUIPOS O PROCEDIMIENTOS

A continuación, se describen las distintas tareas realizadas durante el proyecto y los equipos o programas utilizados durante estas.

- Análisis de las ofertas de módulos de recepción *GPS*.
- Diseño del esquemático:
    - Software de desarrollo *OrCAD Capture*.
- Selección de distribuidor y componentes del diseño esquemático.
- Creación del *BOM*.
- Diseño de las huellas de los componentes:
    - Software de desarrollo *OrCAD Padstack Editor*.
    - Software de desarrollo *OrCAD PCB Editor*.
- Diseño del *PCB*:
    - Software de desarrollo *OrCAD PCB Editor*.
    - Herramienta de diagnóstico *Eurocircuits PCB Configurator*.
    - Herramienta de diagnóstico *Eurocircuits PCB Checker*.
- Revisión del diseño del PCB.
- Producción del *PCB.*
- Adquisición de componentes.
- Montaje de los componentes en la placa prototipo:
    - Estación de soldadura.
- Comprobación del funcionamiento del prototipo:
    - Fuente de alimentación.
    - Osciloscopio digital.
- Implementación de la sincronización:
    - *Raspberry pi 3*.
    - *GPSd*.
    - *pps-tools*.
- Realización de la memoria.
- Revisión y edición.
- Entrega documento.
- Defensa.

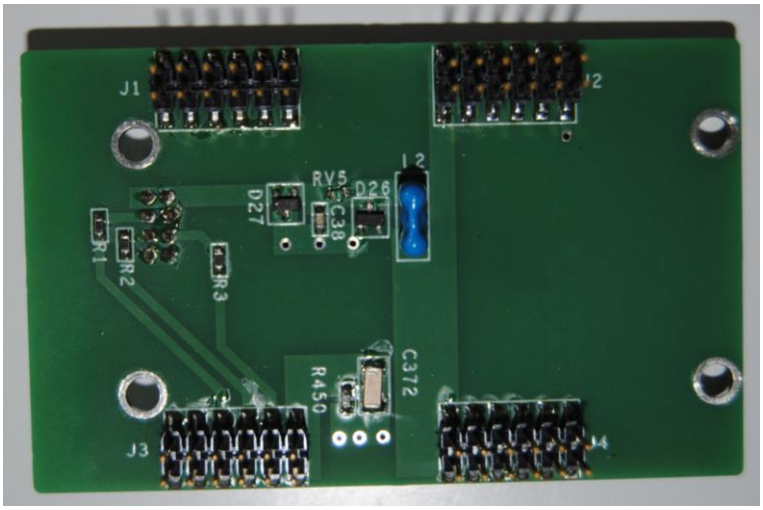
2.2 CRONOGRAMA DE TAREAS

Tareas	2017		2018												2019							
	Nov	Dic	Ene	Feb	Mar	Abr	May	Jun	Jul	Ago	Sep	Oct	Nov	Dic	Ene	Feb	Mar	Abr	May	Jun	Jul	Sep
Análisis de las ofertas de módulos de recepción *GPS*																						
Diseño del esquemático																						
Selección de distribuidor y componentes del diseño esquemático																						
Creación del BOM																						
Diseño de las huellas de los compo-nentes																						
Diseño del PCB																						
Revisión del diseño del PCB																						
Producción del PCB																						
Adquisición de los componentes																						
Montaje de los componentes en la placa prototipo																						
Comprobación del funcionamiento del prototipo																						
Implementación de la sincronización																						
Realización de la memoria																						
Revisión y edición																						
Entrega documentación																						
Defensa																						

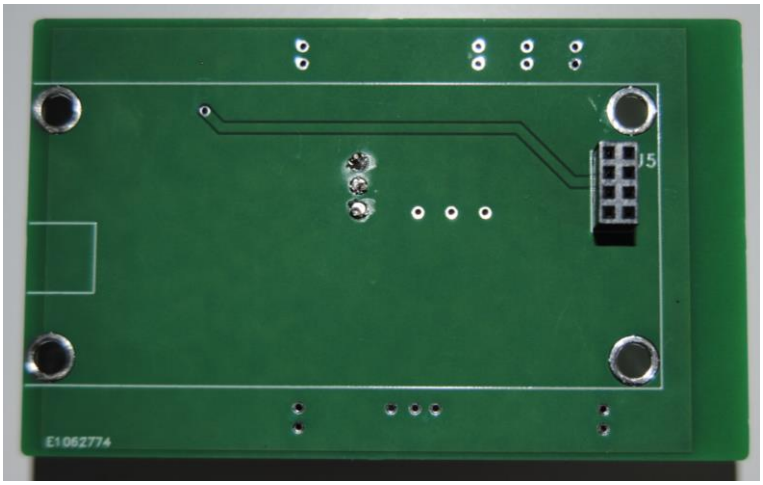
Tabla 3. Cronograma de las tareas realizadas.

2.3 DESCRIPCIÓN DE LOS RESULTADOS

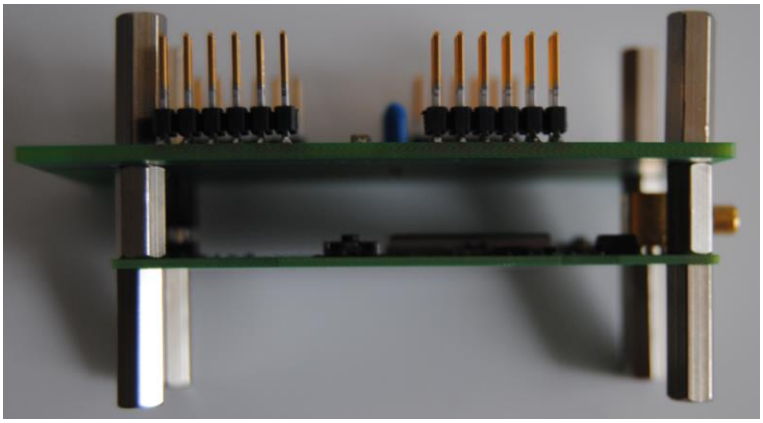
Una vez se reciben los componentes y la primera serie de *PCBs* se empieza con el montaje para poner a punto el prototipo sobre el que se van a realizar las pruebas. Al realizar la soldadura es importante prestar atención a los conectores *J1 – J4* ya que, realizando el montaje a mano y siendo los conectores *SMD*, pueden quedar desplazados originando problemas a la hora de introducir el módulo en uno de los hechos del sistema RELY-CPPS. En cuanto al resto de componentes, solo cabe destacar que la huella del dispositivo *RV5* es demasiado estrecha, pero, a pesar de ello, ha sido posible soldarla. Por lo que para futuras revisiones es recomendable aumentar la separación entre los *pads*.



Captura 18. Tarjeta soldada por la cara inferior, donde se conecta a la plataforma RELY-CPPS.



Captura 19. Tarjeta soldada por la cara superior, donde se conecta el módulo ICM STM 360.



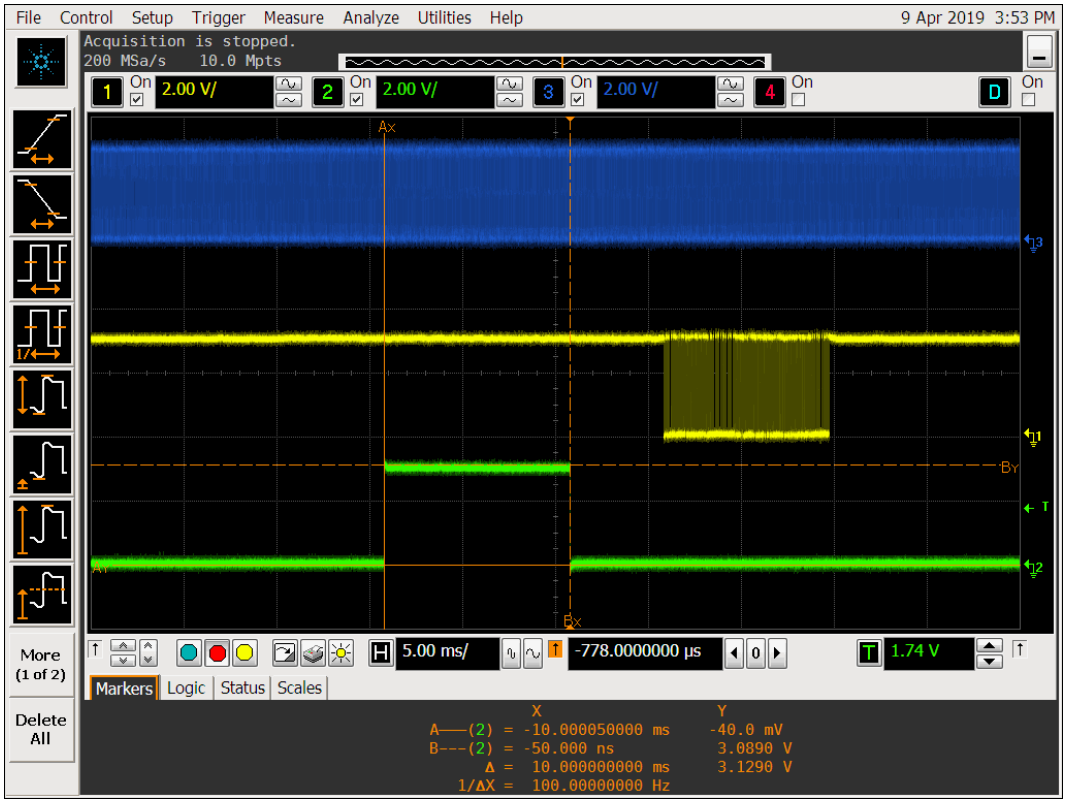
Captura 20. Sistema completo montado.

Con el montaje finalizado, hay que probar que el sistema funcione. El módulo *ICM STM 360* está preparado para buscar de manera automática satélites a los que conectarse. Si se trata de un arranque en frío, es decir, no tiene guardados el almanaque, la hora, el hemisferio o la posición, empieza a conectarse con la red *GNSS* mientras envía mensajes *TSIP* periódicamente. Es necesario que el módulo busque satélites de manera ininterrumpida durante aproximadamente 15 minutos para descargar un almanaque completo. Este almanaque contiene entre otras cosas el tiempo *UTC* que se actualiza cada segundo, por lo que es necesario para poder obtener las señales que se desean.

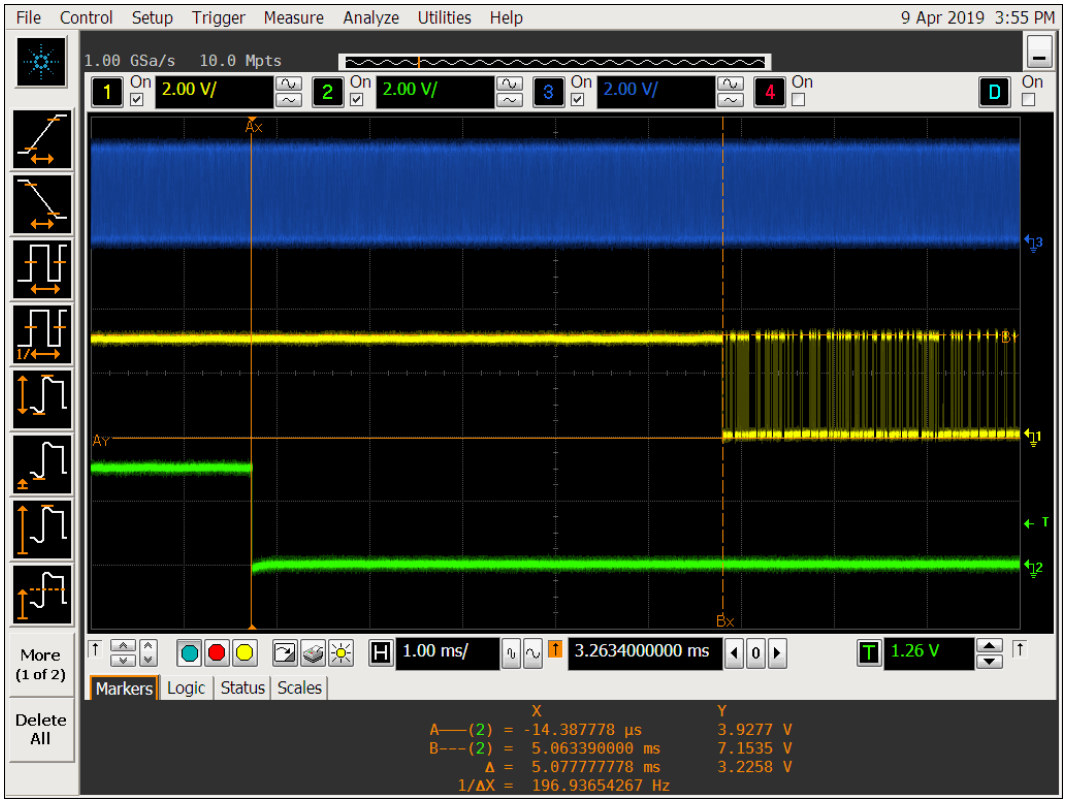
En cambio, una vez ha adquirido y se ha enganchado con los suficientes satélites, empieza a recibir datos sobre la posición y hemisferio. Tras obtener un valor de posición fijo durante varios ciclos el sistema entra en un estado de operación automática en el que transmite las señales *PPS* y *SYSCLK*, además del tiempo *UTC* actualizado por la conexión serie. Toda la comunicación serie es a través del protocolo de comunicación propiedad de *Trimble*: *TSIP*, por lo que la hora está también en este formato. Una vez que se ha realizado el arranque en frio una vez, el tiempo que tarda en entrar en modo automático se ve reducido drásticamente.

Por tanto, si simplemente se alimenta el sistema con una fuente de alimentación y se conecta la antena *GNSS* el *ICM STM 360* entra en modo automático sin necesidad de configurar nada. Con un osciloscopio se pueden capturar las señales importantes para el desarrollo del proyecto: *PPS*, *SYSCLK* y *TXDA*. Es importante dejar en funcionamiento durante un rato el sistema para que el receptor procese los datos que recibe de los satélites para que la señal PPS sea lo más estable y precisa posible, especialmente a la hora de observar la señal para su captura. En funcionamiento normal, cuando tiene la posición fijada, la precisión de la señal es de 15 ns. Sin embargo, la calidad disminuye si se encuentra procesando datos antes de entrar en modo de funcionamiento automático hasta una precisión cercana a 50 ns. Como la implementación que se desarrolla en este trabajo es estática, no es un factor influyente.

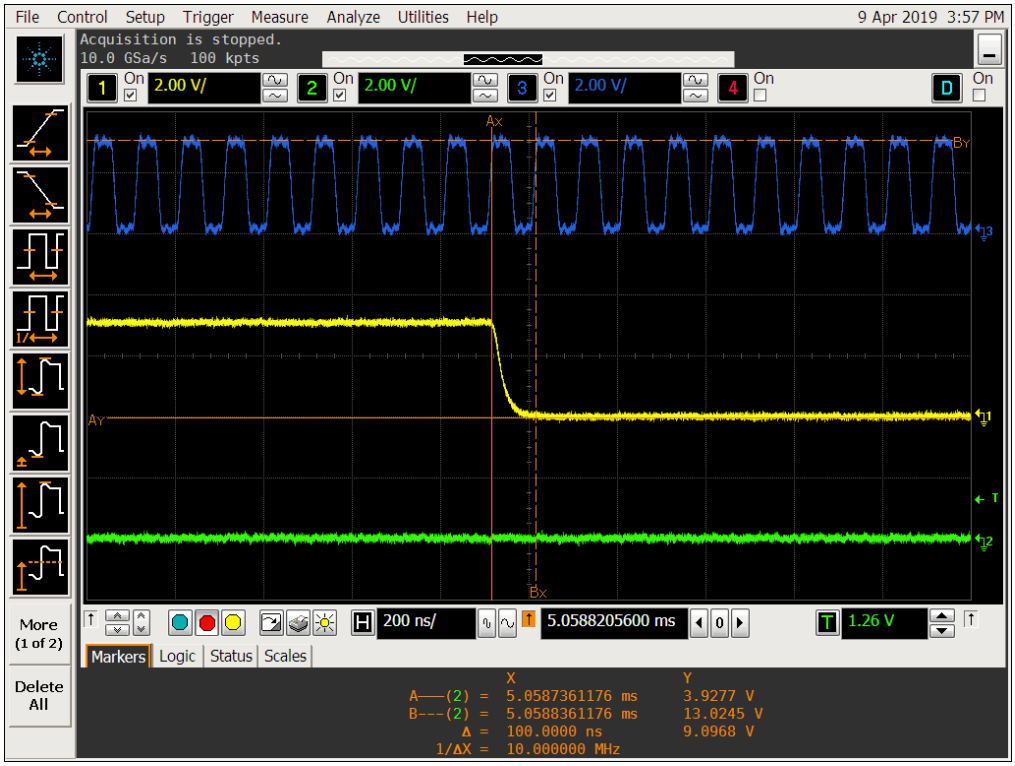
Las señales que se muestran a continuación demuestran que el módulo se encuentra en el modo automático. Se observa como la comunicación empieza con el pulso *PPS* de 10 ms de duración, que tras un retardo de 5 ms se reciben los datos a través de la conexión serie *TXDA*.



Captura 21. Señales generadas por el módulo ICM-STM-360: SYSCLK de 10 MHz (azul), TXD (amarillo), PPS (verde).

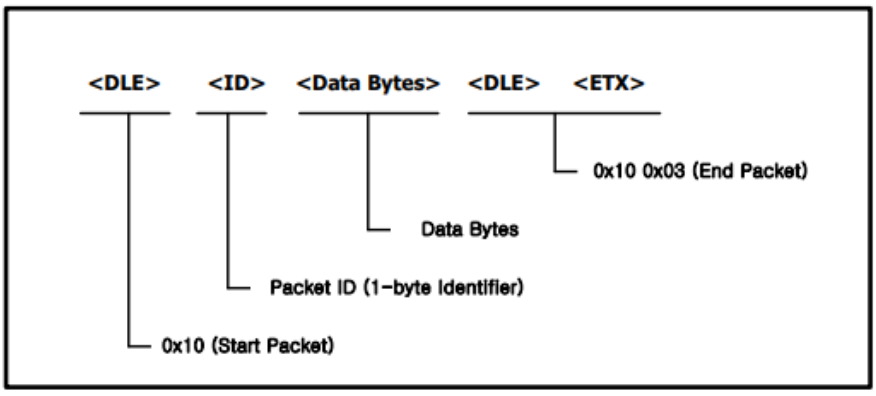


Captura 22. Retardo entre la señal PPS y los datos que transmite el módulo a través de la conexión serie.



Captura 23. Sincronización de la comunicación serie y la señal SYSCLK.

El siguiente paso después de comprobar que el módulo entra en el modo de funcionamiento automático es cambiar el protocolo de salida de la comunicación serie de *TSIP* a *NMEA*. Como se ha mencionado anteriormente, el módulo de *GPS ICM STM 360* tiene la capacidad de configurarse para funcionar haciendo uso del estándar *NMEA*. Existen apartados en el propio manual de usuario que detallan que mensajes se deben enviar para cambiar la configuración de los puertos de entrada/salida. Sin embargo, estos mensajes hacen uso del protocolo *TSIP* que poseen una estructura como la que se muestra a continuación, por lo que es necesario crear un pequeño ejecutable que envíe los mensajes al menos la primera vez que se enciende el sistema para dejarlo en el modo de funcionamiento adecuado.



Captura 24. Formato de los mensajes del protocolo TSIP desarrollado por Trimble.

Donde *<ID>* es el byte que identifica de qué tipo de paquete se trata y los bytes de datos tienen el resto de los parámetros que se especifican en el apartado del paquete que se encuentra detallado en el manual de usuario.

Consultando el capítulo sobre el protocolo *NMEA 0183* del manual de usuario de *Trimble*, se obtienen los 3 tipos de mensajes necesarios para desarrollar la aplicación:

1. *0xBC*: cambiar la configuración del puerto.
2. *0x7A*: cambiar la configuración del protocolo *NMEA*.
3. *0x8E-26*: guardar en la memoria no volátil del sistema la configuración para hacerla permanente.

No obstante, para el desarrollo de este proyecto no es necesario cambiar la configuración del protocolo *NMEA* debido a que la configuración por defecto es suficiente para comprobar el funcionamiento y no se guardarán los cambios de forma permanente en el prototipo. Aunque para la implementación final es imperativo guardar la configuración para no tener que configurar el receptor de GPS ICM STM 360 cada vez que se encienda. Por lo que solo se hace uso del primer mensaje.

Para ello, se emplea una *Raspberry Pi 3* que permite, usando el lenguaje de programación *Python* y el puerto serie que tiene incorporados, enviar los mensajes mencionados anteriormente de manera fácil y sencilla. Además, también permite instalar un programa capaz de interpretar los datos recibidos del *GPS* que, de estar correctamente configurado, los represente de forma entendible. Para ello se emplea el *daemon GPSd* que proporciona una interfaz para receptores de GPS de diversos tipos y permite ser accedido por múltiples aplicaciones al ser un servicio o programa residente que se ejecuta en segundo plano. Adicionalmente, es fácil de instalar e iniciar en la plataforma *Raspberry Pi 3* que se utiliza en este proyecto para comprobar el sincronismo de la aplicación.

El mensaje *0xBC* tiene el siguiente formato:

Byte	Nombre	Valor	Descripción
0	Identificador de paquete	0xBC	Modificar la configuración de los puertos.
1	Puerto	0x00	Puerto 1 (único accesible).
2	Velocidad de transmisión de entrada	0x0B	Se mantiene la velocidad por defecto.
3	Velocidad de transmisión de salida	0x07	Se cambia a 9600 baudios para que concuerde con la velocidad de la *Raspberry Pi 3*.
4	Bits de datos	0x03	Se mantienen 8 bits.
5	Paridad	0x01	Se mantiene paridad impar.
6	Bits de parada	0x00	Se mantiene un bit de parada.
7	Reservado	0x00	Reservado.
8	Protocolo de entrada	0x02	Se mantiene el protocolo *TSIP* para los mensajes de entrada.
9	Protocolo de salida	0x06	Se cambia al protocolo *NMEA* para los mensajes de salida.
10	Reservado	0x00	Reservado.

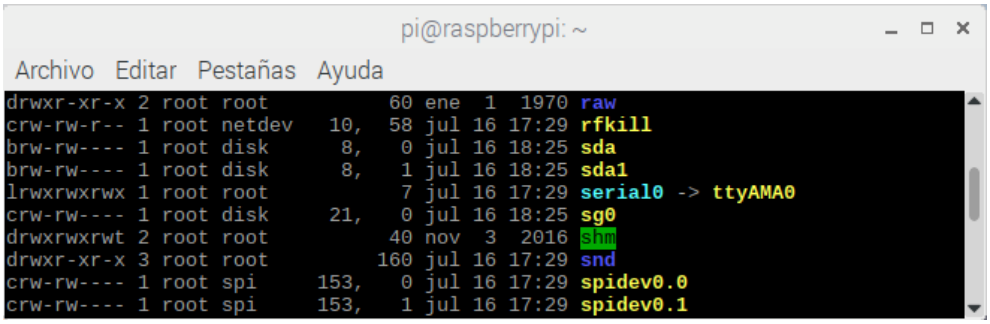
Tabla 4. Valores del mensaje de configuración del puerto serie para la aplicación.

Por lo que añadiendo los identificadores de inicio y final de paquete que se han mostrado se crea el paquete que se debe enviar en el ejecutable (ver Anexo VI) a través del puerto serie de la Raspberry Pi 3. Si se conecta el puerto de la recepción serie de la *Raspberry Pi 3* a la transmisión del sistema una vez se ha ejecutado el programa de configuración del puerto se deben observar los mensajes del protocolo *NMEA*. A pesar de que el torrente de datos sin procesar no es comprensible, sí que tiene un formato característico que de visualizarse escuchando en el puerto de recepción serie es reconocible. Estos se transmiten en formato *ASCII*, así que al no haber modificado los mensajes *NMEA* que envía el receptor de *GPS* se observan los paquetes por defecto.

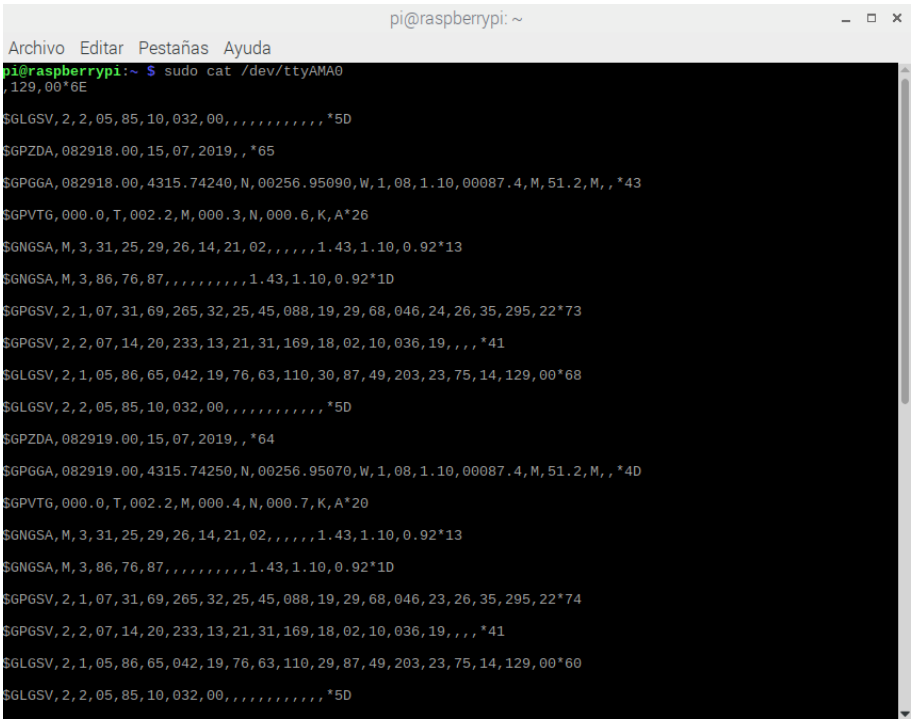
Mensaje	Descripción
GGA	Información de la posición en el que se encuentra el receptor que es esencial para la localización 3D.
GSA	Detalles sobre la naturaleza de la posición. Incluye datos como el número de satélites utilizados para obtener la posición y precisión (en inglés *Dillution Of Precision*, *DOP*).
GSV	Información sobre los satélites que pueden encontrarse en función de la vista del cielo y datos almacenados en el almanaque.
VTG	Información sobre la velocidad.
ZDA	Fecha y hora.

Tabla 5. Paquetes NMEA enviados por defecto por el receptor GPS.

Por tanto, estos campos deben de ser distinguibles para verificar que el receptor de *GPS* ha recibido correctamente el mensaje de configuración y en efecto se está trasmitiendo el torrente de información en formato *NMEA*. El dispositivo por el que se recibe la información del *GPS* es el *ttyAMA0* y, por tanto, es el dispositivo al que se tiene que enganchar el *daemon GPSd* para obtener los datos *NMEA* que debe procesar. El recurso serie debe ser declarado en el fichero */boot/config.txt* (ver Anexo VIII) para que sea accesible tanto para enviar el mensaje del ejecutable como para conectarlo al *daemon*. Una vez se han añadido las líneas correspondientes, se puede reiniciar la *Raspberry Pi 3* y si se ejecuta el comando *"ls -l /dev"* para ver los recursos del sistema, el recurso ttyAMA0 debe aparecer.



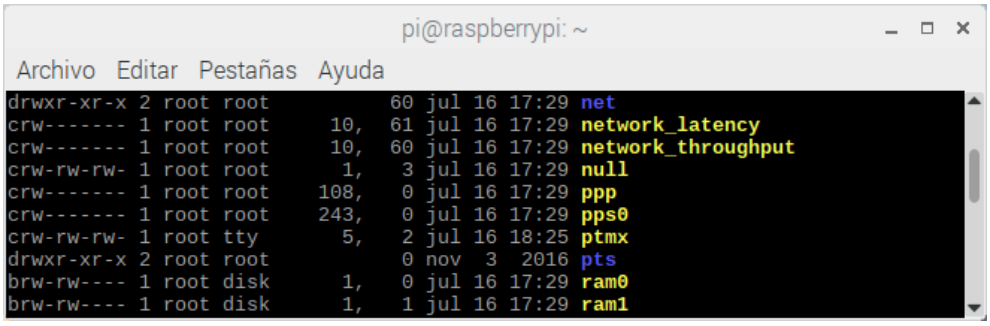
Captura 25. Dispositivo serie ttyAMA0.



Captura 26. Captura de la transmisión de la información del GPS en formato NMEA recibida en puerto serie de la Raspberry Pi 3.

Como se puede observar en la captura superior, los paquetes coinciden con los paquetes por defecto, por lo que el módulo ha recibido el mensaje *0xBC* correctamente y se puede activar el *daemon GPSd* para que escuche en el puerto serie. Si se quisiera añadir o quitar paquetes proporcionados por el protocolo *NMEA* es posible mediante el mensaje de configuración de intervalo y máscara *0x7A*. Este mensaje se construye de la misma forma que el enviado previamente para cambiar el protocolo del puerto de salida (ver Anexo VII).

Una vez que se demuestra que el torrente de datos *NMEA* es recibido correctamente se puede ejecutar el *daemon* de *GPS* enganchado al dispositivo */dev/ttyAMA0.* Para ello hay que especificar que recurso tiene que acceder (ver Anexo IX) e iniciar el programa. De la misma forma que con el dispositivo anterior, el recurso pps0 debe de aparecer disponible.



Captura 27. Dispositivo pps0.

Una vez el daemon empieza a funcionar, la forma de comprobar si los mensajes son interpretados de forma correcta por el servicio es ejecutando el cliente *cgps* que muestra los datos procesados.

Captura 28. Cliente cgps que muestra la información del daemon GPSd de forma visual.

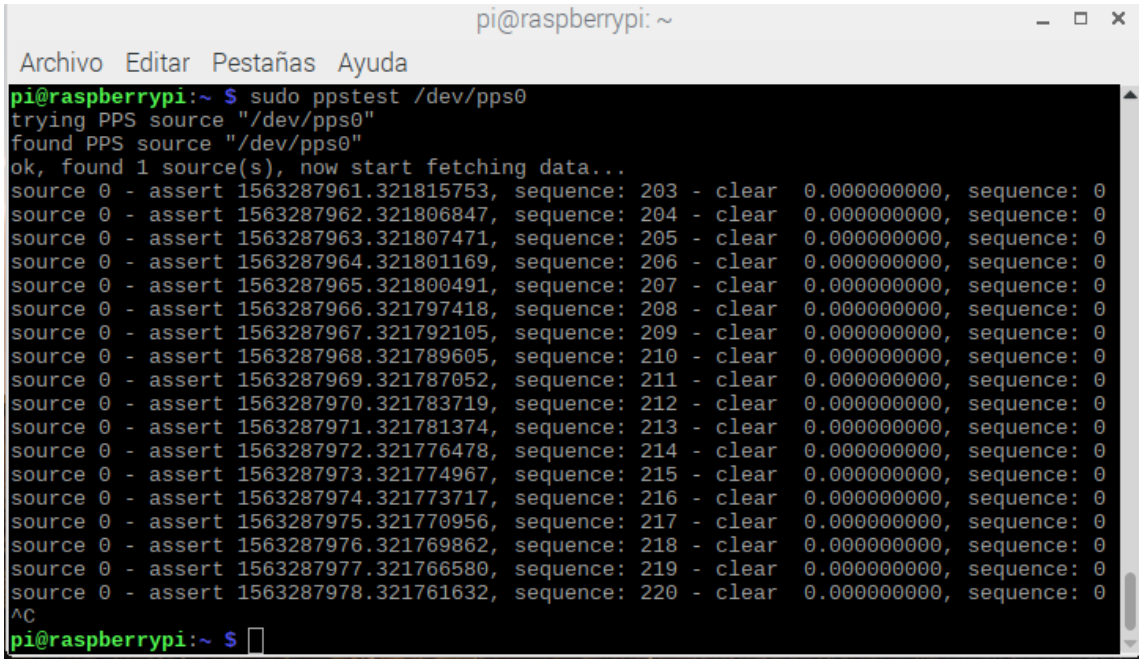
Se puede observar cómo se muestra la hora *UTC* y el offset propio de la zona horaria en la que se encuentra el receptor de *GPS*. Sin embargo, el sistema no se encuentra del todo sincronizado, ya que existe un retardo en la transmisión del torrente de datos NMEA que es enviado antes del inicio del siguiente segundo. Para terminar la aplicación y mejorar el sincronismo se puede enganchar el *daemon* a una señal *PPS* como la que produce el módulo receptor de *GPS*.

El flanco de la señal *PPS* determina el momento exacto en el que empieza un segundo, eliminando variaciones que pueden producirse si se utiliza la recepción de datos NMEA como elemento de sincronización. Esta señal puede conectarse a uno de los pines de propósito general disponibles de la plataforma *Raspberry Pi 3* y especificar el pin al *daemon GPSd*.

Existe una herramienta disponible para el manejo de una señal de PPS que funciona de manera similar al *daemon* de *GPS* denominada *pps-tools*. Simplemente se instala en el sistema y se especifica que la señal *PPS* se conecta al pin de propósito general *GPIO4*. Es necesario especificar al sistema que el recurso se conecta en el pin de propósito general (ver Anexo VIII). De tal forma que el sistema reconoce que el dispositivo */dev/pps0* existe y esta referenciado al *GPIO4*.

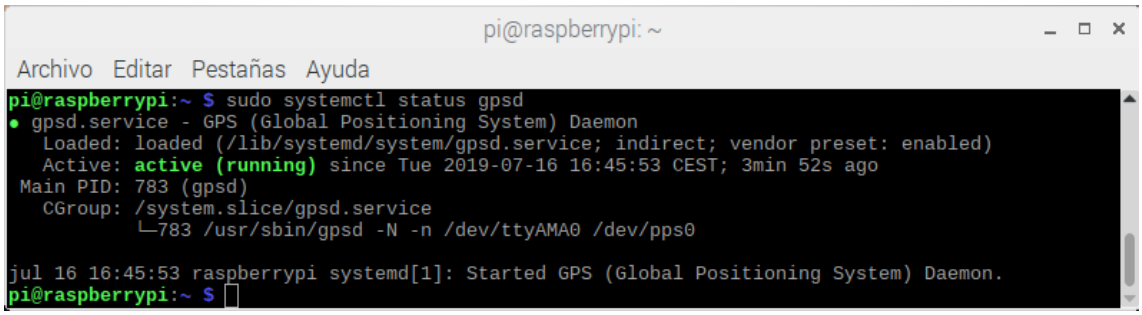
Además, la herramienta incluye la capacidad de realizar una prueba escuchando en el dispositivo creado previamente, por lo que de funcionar correctamente debe visualizarse una fuente que es la señal PPS que se actualiza cada segundo.

Como se puede observar en la siguiente captura del ensayo, se ha encontrado una fuente *PPS* en el dispositivo *pps0* y se va actualizando cada segundo, demostrando que la herramienta se ha enganchado correctamente.



Captura 29. Prueba del funcionamiento del dispositivo receptor de la señal PPS.

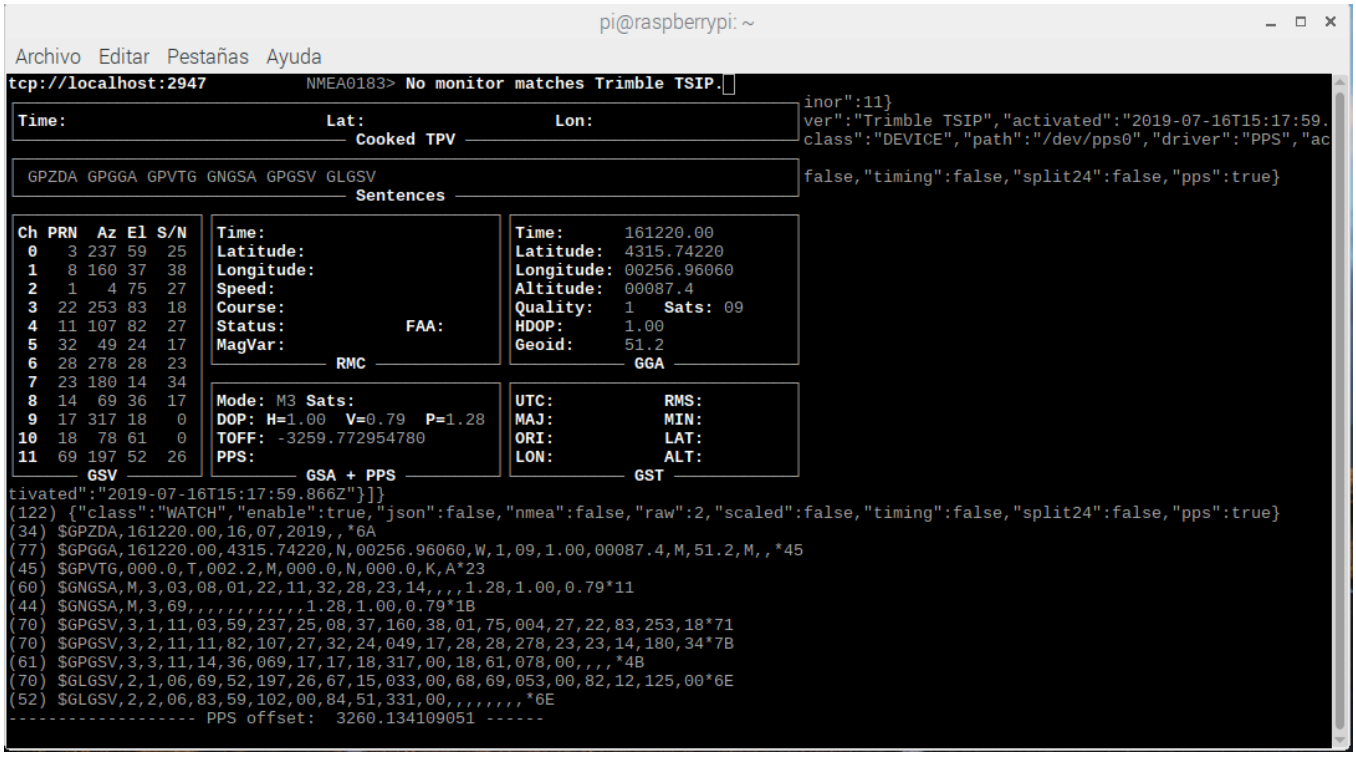
Finalmente, si el dispositivo recibe la señal *PPS*, simplemente hay que especificar al daemon *GPSd* que además del puerto serie por el que recibe el torrente de datos *NMEA*, también debe escuchar el dispositivo *pps0*. Para conectar el recurso al daemon basta con añadir el nuevo dispositivo al comando de inicialización de la aplicación: *"gpsd /dev/ttyS0 /dev/pps0 -F /var/run/gpsd.sock"*. Si la señal ha sido enganchada correctamente debe aparecer el *PPS* como recurso adicional aparte del *ttyAMA0* al comprobar el estado del *GPSd*.



Captura 30. Estado del daemon de GPS con la señal de PPS enganchada.

Como se puede observar, el daemon está activo y funcionando y dispone de dos recursos: la conexión serie *ttyAMA0* a través de la cual se reciben los datos *NMEA* y la conexión pps0 que recibe la señal *PPS*.

Otra forma comprobar que la señal PPS se encuentra enganchada correctamente y el daemon la interpreta correctamente es ejecutar otro cliente más complejo que el utilizado previamente. El *gpsmon* muestra la señal *PPS* con un offset prácticamente idéntico al obtenido mediante el torrente de datos *NMEA*. Por tanto, se demuestra que el GPSd se encuentra sincronizado correctamente.



Captura 31. Cliente gpsmon que muestra la información del daemon GPSd de forma visual.

2.4 PRUEBAS DE ESTABILIDAD

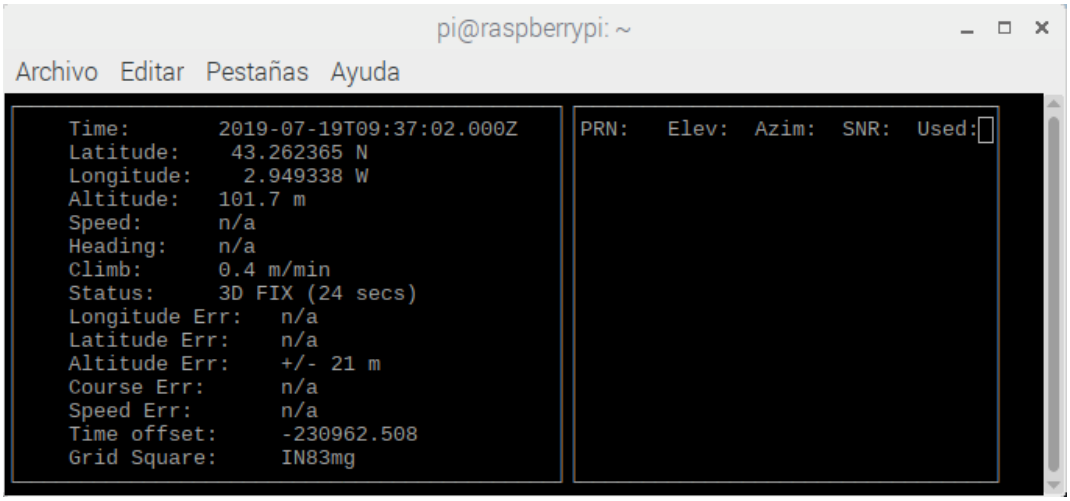
Hasta ahora se ha demostrado la funcionalidad de la aplicación, es decir, se ha visto cómo conseguir sincronizar el *daemon* e *GPS* instalado en la *Raspberry Pi 3*. A pesar de ello, es igual de importante estudiar los fallos que pueden producirse sobre el funcionamiento del sistema para prevenir e intentar reducir las repercusiones que pueden ocasionar. Para ello se propone realizar los siguientes ensayos para estudiar la estabilidad del sistema sincronizado.

Se suponen los siguientes escenarios:

- El torrente de datos NMEA falla, pero se mantiene la señal PPS.
- Se pierde la señal PPS, pero se mantienen los datos NMEA.

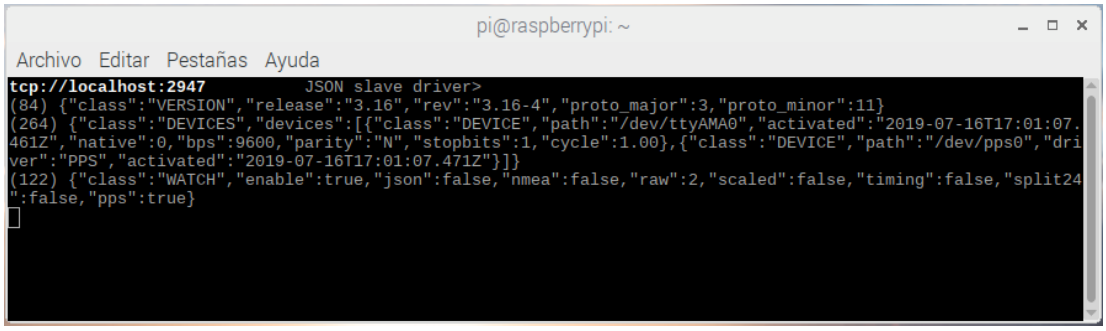
2.4.1 El torrente de datos NMEA falla, pero se mantiene la señal PPS

Para simular este escenario se desconecta la conexión que une el dispositivo *ttyAMA0* con la transmisión serie del módulo receptor de *GPS*. Para observar los efectos que causa perder la información del satélite se ejecutan los clientes usados previamente.



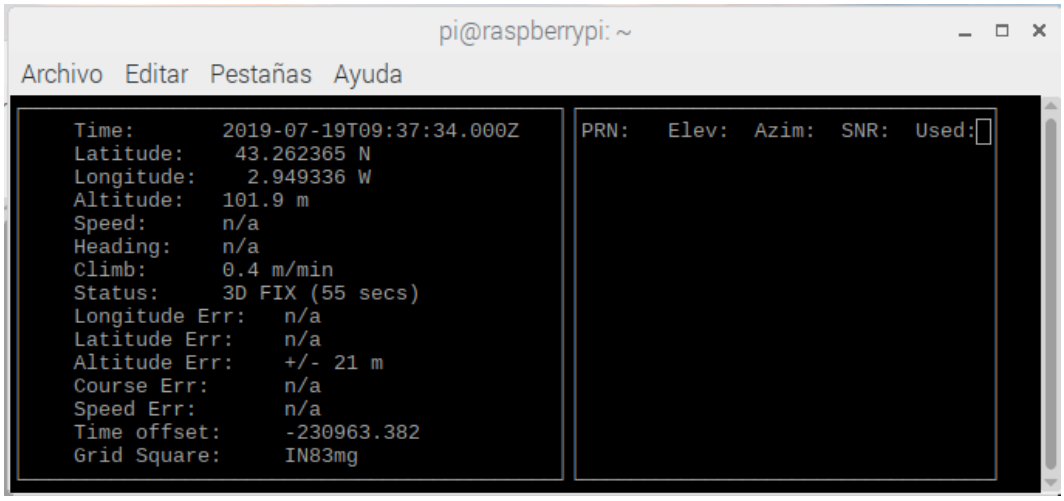
Captura 32. Cliente cgps que muestra la información procesada del daemon GPSd sin torrente de datos NMEA.

El cliente *cgps* no es capaz de seguir actualizando la hora y se queda bloqueado. Obviamente sin datos *NMEA* el cliente no puede funcionar, pero el cliente *gpsmon* es más complejo y sí es capaz de procesar una señal PPS. Sin embargo, cuando se ejecuta, a pesar de que detecta la señal *PPS* activa (*"pps": true*) no es capaz de arrancar y se queda en un estado de espera a información del satélite a través del dispositivo serie.



Captura 33. Cliente gpsmon que muestra la información del daemon GPSd de forma visual sin torrente de datos NMEA, pero con señal PPS activa.

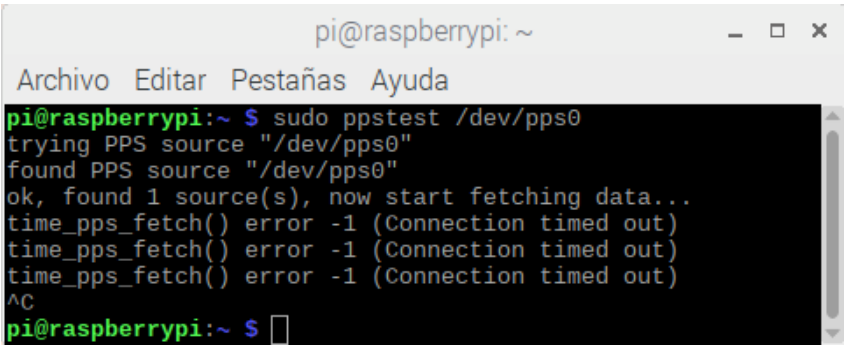
Por lo que de fallar la transmisión de la información *NMEA* el sistema deja de funcionar por completo. No obstante, si el fallo es momentáneo y se reestablece la conexión el *GPSd* vuelve a engancharse sin necesidad de reiniciarse. En cuanto vuelve a estar funcionando el tiempo *UTC* se actualiza



Captura 34. Cliente cgps que muestra la información procesada del daemon GPSd con el torrente de datos NMEA reestablecido.

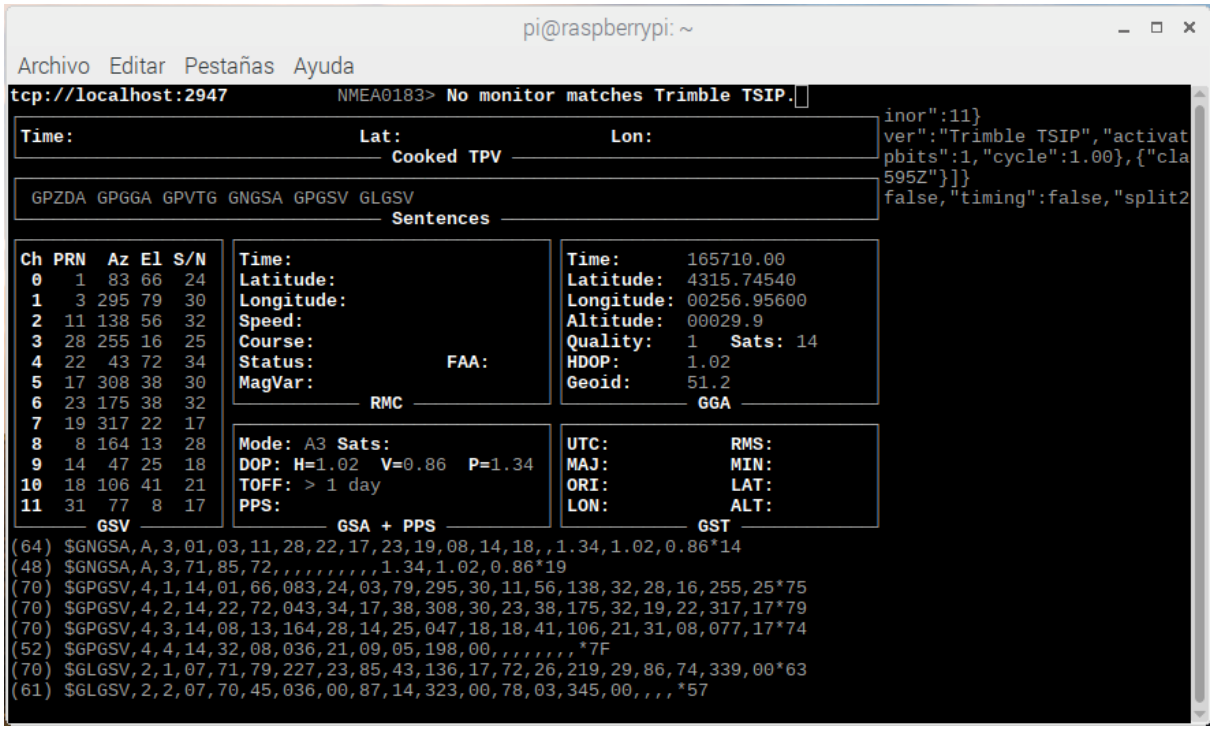
2.4.2 Se pierde la señal PPS, pero se mantienen los datos NMEA

Al igual que en el ensayo anterior, se desconecta la conexión que une el dispositivo *pps0* con la salida de la señal *PPS* del módulo receptor de *GPS*. En este caso sólo el cliente *gpsmon* es capaz de sincronizarse con la señal, por lo que no tiene sentido verificar el cliente *cgps*. Pero también puede testearse la señal usando las herramientas instaladas con el paquete *pps-tools*.



Captura 35. Prueba de la falta de la señal de la señal PPS.

Se aprecia cómo se encuentra el dispositivo *pps0*, pero no recibe ninguna señal y se acaba el temporizador de espera. En cambio, el cliente gpsmon funciona correctamente procesando la información del satélite, pero no detecta ninguna señal *PPS*. Por tanto, se encuentra en el estado anterior de la implementación en la que no está correctamente sincronizado al segundo. Dependiendo de la precisión que requiera la aplicación en la que utiliza el módulo puede funcionar en este estado durante un periodo de tiempo. Si la sincronización que debe tener una aplicación es en cambio, muy precisa, el sistema presenta un fallo grave.



Captura 36. Cliente gpsmon que muestra la información del daemon GPSd de forma visual con torrente de datos NMEA, pero con señal PPS inactiva.

2.5 PRUEBAS ADICIONALES

Aparte de los ensayos realizados, es interesante realizar pruebas para comprobar la estabilidad del sistema, pero para ello es necesario disponer de un receptor GPS adicional para comparar resultados. Sin embargo, en el desarrollo de este proyecto no se dispone de un segundo receptor para llevar a cabo las pruebas, por lo que queda fuera del alcance.

El ensayo pretende comprobar qée ocurre si la calidad de la señal de los satélites disminuye, por ejemplo, metiendo la antena dentro del edificio para que no tenga una vista completa del cielo o desconectado la antena del receptor.

A continuación, se conecta al *daemon GPSd* y se contrastan los efectos que produce con su contraparte que está funcionando correctamente. La importancia reside en verificar si la señal PPS deriva respecto a la que sigue conectada a la red de satélites. Si esta es capaz de mantener el sincronismo durante algunos instantes, significa que cortes momentáneos no afectan al servidor *NTP/PTP*, mientras que si se desincroniza rápidamente puede suponer un problema serio.

3 ASPECTOS ECONÓMICOS

3.1 DESCRIPCIÓN DEL PRESUPUESTO EJECUTADO

3.1.1 Presupuesto de los componentes adquiridos:

Componente	Cantidad	Precio Total
EMI103T-RC, L2	5	2.37 €
SM712.TCT, D27	5	9.30 €
SQT-104-01-L-D, J5	2	3.02 €
ERJ3EKF1001V, R1, R2, R3	10	0.492 €
RC0603FR-072K2L, R450	10	0.293 €
302R29W102KV4E, C372	10	8.71 €
0ZCM0020FF2G, RV5	10	1.65 €
GRM188R61A106ME69D, C38	10	1.74 €
BZX84C3V3, D26	5	0.705
TSM-106-02-S-DV, J1, J2, J3, J4	8	18.56 €
	Total Componentes	46.84 €
	IVA (21%)	9.84 €
	Total	56.68 €

Tabla 6. Presupuesto de los componentes adquiridos.

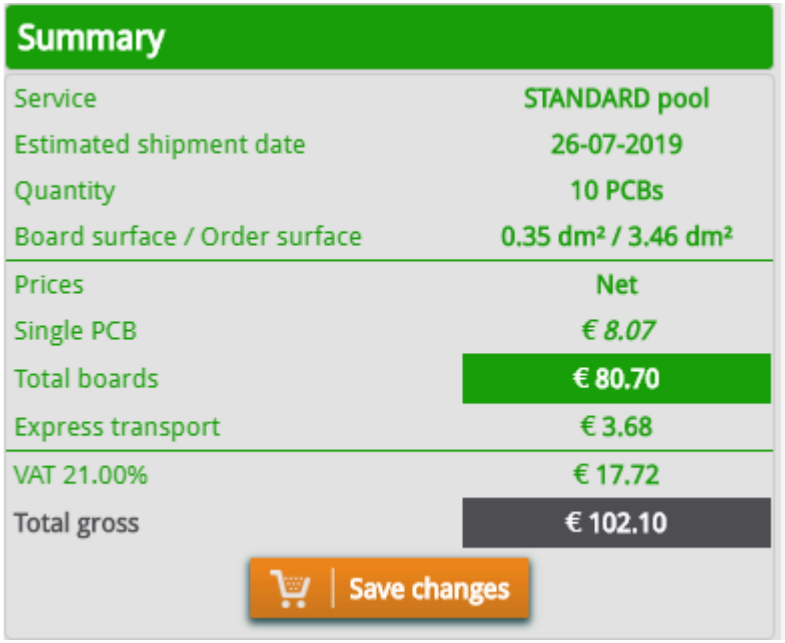
3.1.2 Presupuesto del módulo receptor de GPS ICM STM 360:

Componente		Precio
ICM STM 360		91 €
Antena *GPS*		10.6 €
	Total Componentes	101.6 €
	IVA (21%)	21.34 €
	Total	122.94 €

Tabla 7. Presupuesto del módulo ICM STM 360 y la antena necesaria para su funcionamiento.

3.1.3 Presupuesto de la producción del PCB:

Eurocircuits dispone también de una calculadora de precios para la producción de tarjetas. Como se han subido los *Gerbers* a la aplicación que permite detectar errores, pueden usarse también para obtener el precio que va a costar producirlas. Para este proyecto se han producido 10 unidades de PCB y se elige la estimación de la entrega en 7 días laborables se obtiene el siguiente presupuesto.



Captura 37. Presupuesto obtenido a través de la calculadora de Eurocircuits para 10 tarjetas.

3.1.4 Presupuesto total del proyecto:

Parte		Precio
Componentes electrónicos		56.68 €
Receptor de *GPS*		122.94 €
PCB		102.10 €
	Total Proyecto	281.72 €

Tabla 8. Presupuesto total del proyecto.

3.2 ANÁLISIS DE RENTABILIDAD

A la hora de analizar la viabilidad económica de un proyecto que se basa en la producción de tarjetas y adquisición de componentes es importante tener en cuenta que el coste por tarjeta varía según la cantidad de *PCBs* o de componentes que se adquieran de una vez. Por tanto, elegir una tirada de tarjetas grande para abaratar el presupuesto de la producción es esencial.

También hay que tener en cuenta que no se ha reducido el precio del receptor de GPS debido a que no se adquieren de n distribuidor normal, si no del propio fabricante. Probablemente al comprar grandes cantidades se incluya una deducción de coste por unidad, pero no se considera a la hora de analizar la viabilidad en este proyecto. Tampoco se añade el coste de la antena porque se entiende que es responsabilidad del cliente obtenerla.

Si tomamos por ejemplo el coste de producir 1.000 tarjetas obtenemos, con una fecha estimada de entrega de 12 días laborables, el siguiente presupuesto para la adquisición de materiales.

Referencia	Precio por unidad (1000+)	Unidades por Tarjeta	Precio Total por Tarjeta
C38	0.0406 €	1	0.0406 €
C372	0.346 €	1	0.346 €
D26	0.141 €	1	0.141 €
D27	0.693 €	1	0.693 €
J1,J2,J3,J4	1.08 €	4	4.32 €
J5	0.979 €	1	0.979 €
L2	0.236 €	1	0.236 €
RV5	0.152 €	1	0.152 €
R1,R2,R3	0.0141 €	3	0.0423 €
R450	0.0108 €	1	0.0108 €
ICM STM 360	91 €	1	91 €
PCB	1.02 €	1	1.02 €
	Total		98.981 €
	IVA (21%)		20.786 €
	Total por Tarjeta		119.767 €

Tabla 9. Coste de adquisición de los materiales por tarjeta para una tirada de 1000 unidades.

4 CONCLUSIONES

En este proyecto se ha realizado el desarrollo de un módulo de sincronización para la plataforma *RELY-CPPS*. Para ello se ha realizado un estudio del mercado para seleccionar el receptor *GPS* más adecuado para la implementación, se ha diseñado la tarjeta para implementar el receptor y demostrado que es posible sincronizar un sistema externo enganchado al torrente de datos *NMEA* y señal *PPS* que proporciona el receptor seleccionado. Las pruebas finales desarrolladas en el Capítulo 2.3 demuestran que se cumplen los requisitos necesarios para convertir un sistema que incluya soporte para los protocolos *NTP/PTP,* como lo es la plataforma RELY-CPPS, en un servidor haciendo uso de estos protocolos estandarizados y convertirse en un reloj maestro.

Por lo que se concluye que el objetivo inicial de dotar a la plataforma *RELY-CPPS* de un módulo que implemente el hardware necesario para convertirse en un servidor *NTP* o *PTP* es posible de implementar gracias a la tarjeta diseñada. Sin embargo, como esta parte de la aplicación debe desarrollarse sobre el propio sistema *RELY-CPPS*, queda fuera del alcance propuesto al inicio del trabajo.

Tras el análisis sobre el coste de la producción para el prototipo diseñado, se demuestra que el precio de la producción del módulo es lo suficientemente asequible como para ser un producto rentable. No obstante, para futuras revisiones es importante revisar la huella del componente *RV5* que en el montaje ha resultado difícil soldarlo al ser la huella del estándar *0603*.

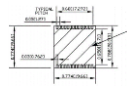
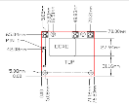
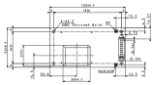
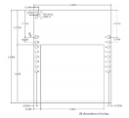
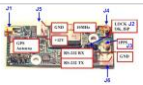
Finalmente, hay que tener en cuenta que por la necesidad de simplificar el prototipo se ha elegido el receptor de *GPS ICM STM 360* implementado en una tarjeta, pero existe la opción de adquirir el receptor en formato *SoC* sin los circuitos añadidos que tiene implementado en la tarjeta. Esta solución reduciría el tamaño del módulo, pero dificultaría el diseño de este al tener que incluir más componentes. Por tanto, es recomendable para un diseño final.

5 BIBLIOGRAFÍA

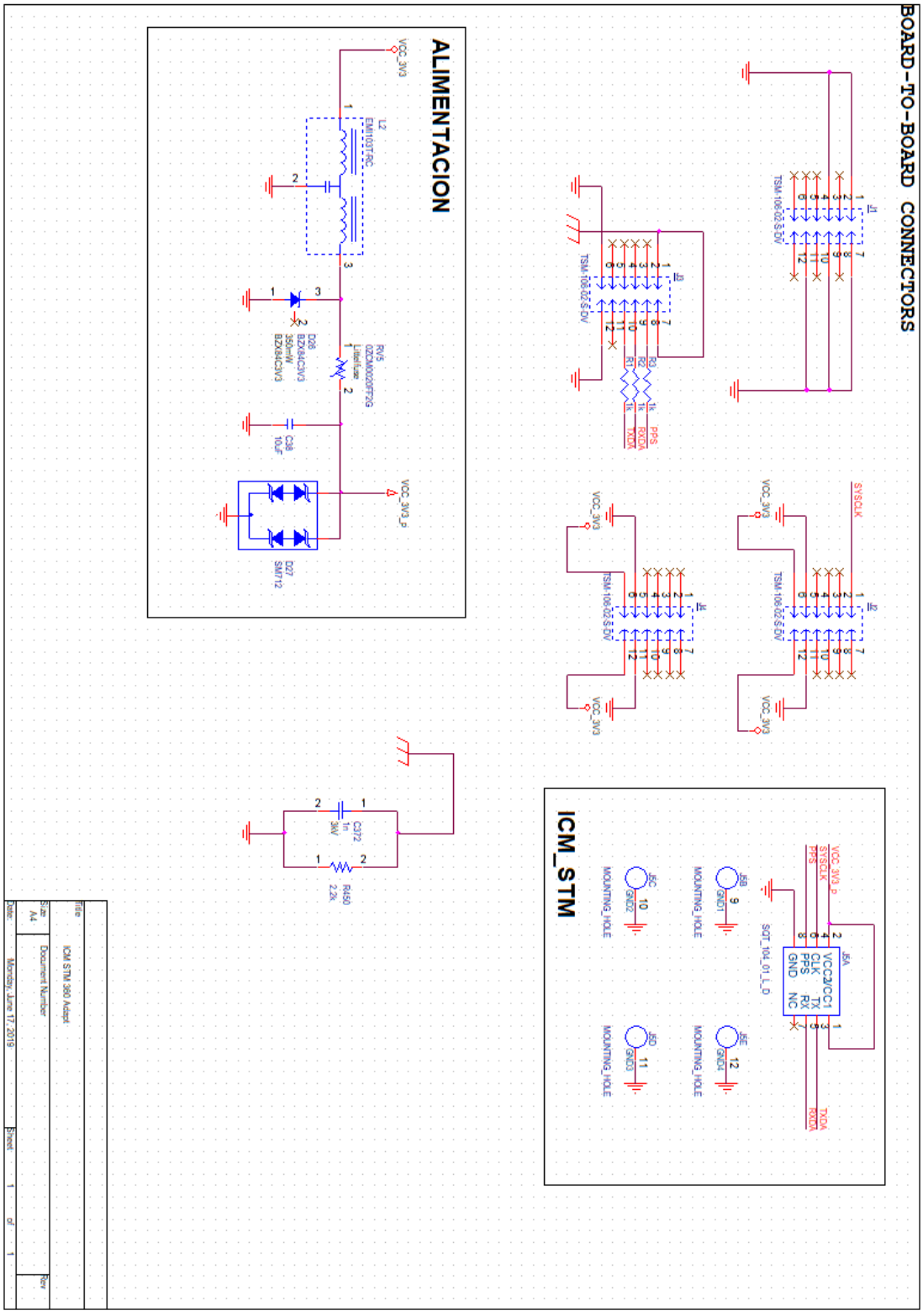
- https://www.relyum.com/web/rely-cpps/
- https://www.trimble.com/Timing/ICM-SMT-360.aspx
- https://www.tronico.fi/OH6NT/docs/NMEA0183.pdf
- https://www.masterclock.com/support/suggested-functional-specifications-gps-ntp-poe
- https://www.masterclock.com/company/masterclock-inc-blog/network-time-synchronization-why-you-need-an-ntp-server
- https://www.masterclock.com/support/library/network-timing-ntp-vs-ptp
- https://gpsd.gitlab.io/gpsd/index.html
- http://wiki.dragino.com/index.php?title=Getting_GPS_to_work_on_Raspberry_Pi_3_Model_B
- http://www.unixwiz.net/techtips/raspberry-pi3-gps-time.html

6 ANEXOS

6.1 ANEXO I: ESTUDIO DEL MERCADO

	Producto	Alimentación (Voc)	Consumo (W)	Temperatura (ºC)	Dimensiones WxLxH (mm)	PPS Duty Cycle	PPS Jitter	PPS Niveles	Nº	Puertos serie Niveles	Salidas (MHz)	Plano	Plano Consideraciones	Circuitos adicionales	Precios Farnell	Precios Digikey	Precios Mouser	Comentarios
Trimble	Resolution SMT	3,0 - 3,6	0,33	-40 to +85	19x19x2,54	10 us (fixed)	15ns	3,0V CMOS	2	3,3V CMOS	x		Implementación en un PCB más sencilla al ser un circuito integrado.	Alimentación de la antena (ejemplos en user's guide)	x	x	x	Tienen la opción del chip ya montado en una placa(?)
	ICM SMT 360	3,3 ± 5%	0,5	-40 to +85	19x19x2,54		15ns	3,3V LVTTL	2	3,3V LVTTL/CMOS	10 MHz (3,3V LVTTL)				x	x	x	
	RES SMT 360	3,3 ± 5%	0,5	-40 to +85	19x19x2,54		15ns	3,3V LVTTL	2	3,3V LVTTL/CMOS	x				x	x	x	
	Mini-T	5,5 ± 5%	3	-40 to +85	70x76,2x17	x	15ns	3,3V LVTTL	1	3,3V LVTTL/CMOS	10 MHz (LVTTL)			x	x	x	x	
Furuno	GF-8701	3,7 @ ±5,5%	0,555	-40 to +85	34x27x8	50% (typ) programable	15ns	5V tolerant gate	1		10 MHz (5V tolerant gate) 4kHz - 40MHz (5V tolerant gate)		El montaje superficial no es posible. Usa pines through hole.	x	x	x	x	
	GF-8702	3,7 @ ±5,5%	1,665	-40 to +85	34x27x12,5		15ns								x	x	x	
	GF-8703	3,7 @ ±5,5%	2,22	-40 to +85	34x27x17		15ns								x	x	x	
	GF-8704	5,5 @ ±5,5%	2,2	-40 to +85	52x100x20		15ns		1		10 MHz (5V tolerant gate) 4kHz - 40MHz (5V tolerant gate) 10 MHz (analog MMCX/50Ω)			x	x	x	x	
	GF-8705	5,5 @ ±5,5%	2,2	-40 to +85	52x100x20		15ns								x	x	x	
Microsemi	GPS-500	12 @ ±5%	2,4	-40 to +85	40,64x48,26x13,54	x	35ns	LVDS +/-300mV differential 3.3V LVCMOS	1	RS-232 115.2 8N1	-10 MHz (LVDS) 10 MHz (LVDS)		El montaje superficial no es posible. Usa pines through hole.	x	x	x	x	
	GPS-1000	8 - 14	1,3	0 to +60	25,4x63,5x12,7	x	50ns	3.3V CMOS Rising Edge Synchronized	1	115Kbaud 8N1 RS-232	10 MHz (Center-Sine Output, Shield-GND)		Conexiones separadas.	x	x	x	x	
	GPS-2000	12 @ ±5%	3,2	0 to +75	38,1x76,2x16,51	x	50ns	LVDS	1	RS-232	10 MHz			x	x	x	x	No tiene manual y la información del datasheet es escasa.
	GPS-2500	12 @ ±5%	4	0 to +75	38,1x76,2x20,32	x	30ns	LVDS RS-232	1	RS-232	10 MHz			x	x	x	x	No tiene manual y la información del datasheet es escasa.
	GPS-2550			-25 to +75		x									x	x	x	
	GPS-2600	11 - 16	4	-25 to +75	38,1x101,6x20,32	x	30ns	5Vpp CMOS, Rising Edge Aligned, <2ns risetime (Out) LVDS, +/-300mV differential (Output A) RS-232 (Output B)	1	RS-232 115.2 8N1	10 MHz (LVDS +/-300mV differential)		Muchas salidas usan conectores tipo coaxial.	x	x	x	x		
	GPS-2650			-25 to +85		x									x	x	x	
	GPS-2700	8 - 36 (5 via mini-USB)	1,4	-10 to 70	63,5x76,2x17,78	x	15ns	5V CMOS	1	RS-232	5 MHz (5V CMOS, Center:RF-Shield: GND) 4 x 10 MHz (Sine wave, Center:RF-Shield: GND)		Muchas salidas usan conectores tipo coaxial.	x	x	x	x	

6.2 ANEXO II: DISEÑO ESQUEMÁTICO



6.3 ANEXO III: BILL OF MATERIALS

Item	Canti-dad	Referencia	Valor	Referencia Fabricante	Fabricante	Encapsu-lado	Referencia Distribuidor (Farnel)
1	1	C38	10uF	GRM188R61A106ME69D	MURATA	603	2456110
2	1	C372	1n	302R29W102KV4E	JOHANSON DIELECTRICS	1808	1886070
3	1	D26	BZX84C3V3	BZX84C3V3	ON SEMICONDUCTOR	SOT23	1651563
4	1	D27	SM712	SM712-02HTG	SEMTECH	SOT23	1456395
5	4	J1,J2,J3,J4	TSM-106-02-S-DV	TSM-106-02-S-DV	SAMTEC	SMD	2055858
6	1	J5	SQT-104-01-L-D	SQT-104-01-L-D	SAMTEC	INSERCION	1668153
7	1	L2	EMI103T-RC	EMI103T-RC	BOURNS INC.	INSERCION	2858886
8	1	RV5	0ZCM0020FF2G	0ZCM0020FF2G	BEL FUSE INC.	SMD	2834931
9	3	R1,R2,R3	1k	ERJ3EKF1001V	PANASONIC ELECTRONIC	603	2303145
10	1	R450	2.2k	RC0603FR-072K2L	YAGEO	603	9238522

Tabla 10. Bill of Materials.

6.4 ANEXO IV: TECNOLOGÍA DE MONTAJE SUPERFICIAL, *SMT*:

La Tecnología de Montaje Superficial (más conocida como *SMT* del inglés *Surface Mount Technology*) es el método de producción de circuitos electrónicos en el que los componentes son directamente montados en la superficie de las tarjetas de circuito impreso o *PCBs*. Un dispositivo electrónico que es soldado de esta forma es referido como Dispositivo de Montaje Superficial o *SMD*.

Por lo que la huella del componente tiene tantos *pads* como necesite el encapsulado rodeados de una zona libre de cobre conductor que muestra la capa inferior del dieléctrico (generalmente amarillo) que se utiliza para separar las capas conductoras apiladas de la tarjeta. Fuera de esta zona y únicamente en las capas exteriores de la tarjeta se sitúa la máscara de soldadura. Se trata de una capa de lacado que reciben las capas de cobre tanto superior como inferior que evita que se produzcan cortocircuitos y rechaza el estaño. Esta capa es normalmente de color verde, aunque puede cambiarse el color por motivos estéticos.

Captura 38. Captura de la previsualización de los pads del componente D26.

Esta tecnología ha reemplazado a lo largo de la industria electrónica a la tecnología de agujero pasante (en inglés *Through-Hole*, *TH*), en la que los componentes son introducidos en agujeros que atraviesan la totalidad de la tarjeta. A pesar de que ambas tecnologías son empleadas hoy en día generalmente de manera simultánea como en la tarjeta de este proyecto, los componentes *SMD* presentan numerosas ventajas frente a sus contrapartes *TH*.

- Son encapsulados más pequeños.
- Tienen mayor densidad de componentes y suportan más pines por encapsulado.
- Facilitan una mayor densidad de conexiones en la tarjeta. Al no tener agujeros que atraviesen toda la placa, liberan las capas inferiores para que pasen líneas de conexión.
- Son más baratos que sus versiones *TH*.

- Son soldables mediante equipos de montaje superficial automáticos que agilizan enormemente la producción de tiradas grandes de placas. Además, la tensión superficial del estaño que se usa para soldar suele corregir errores de posicionamiento al atraer los componentes hacia los *pads*.
- Un *PCB* repleto de partes *SMD* tiene un perfil más bajo, es más compacto, pesa menos y permite que diseños mucho más complejos quepan en una única tarjeta.

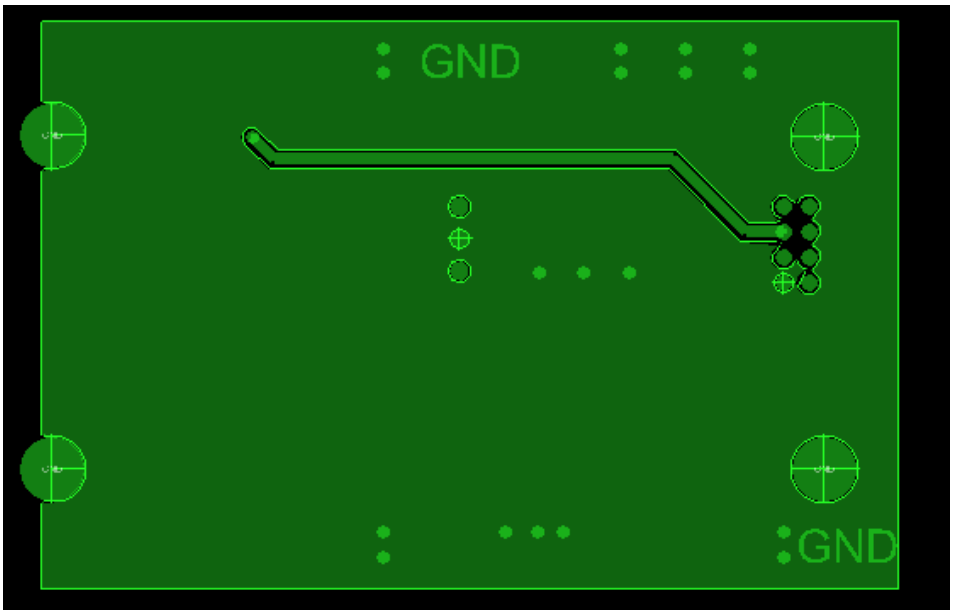
No obstante, no todos los componentes de una tarjeta pueden ser *SMD*. En algunos casos concretos elegir una pieza de agujero pasante es necesario o presenta ciertas ventajas.

- La inmensa mayoría de componentes de alta potencia son *TH*.
- Las partes que sufren estrés mecánico durante su uso como los conectores son de agujero pasante o tienen algún refuerzo que emplee esta tecnología para evitar ser arrancados al ser conectados y desconectados con normalidad.
- Aplicaciones o prototipos que requieran montaje manual o reparaciones son más difíciles si se emplean componentes de *SMD*, especialmente si los componentes son pequeños o tienen muchos *pads*.

Por tanto, para la producción del prototipo de este proyecto se llega al compromiso de elegir todos los componentes posibles en su versión *SMD*, salvo para el conector *J5* (al ser un conector sufre mucho estrés mecánico) y la inductancia *L2* (no existe versión del componente en *SMD*). Sin embargo, se eligen encapsulados *SMD* para facilitar el montaje manual como por ejemplo de tamaño *0603*, en el caso de condensadores y resistencias, y *SOT-23*, para los diodos.

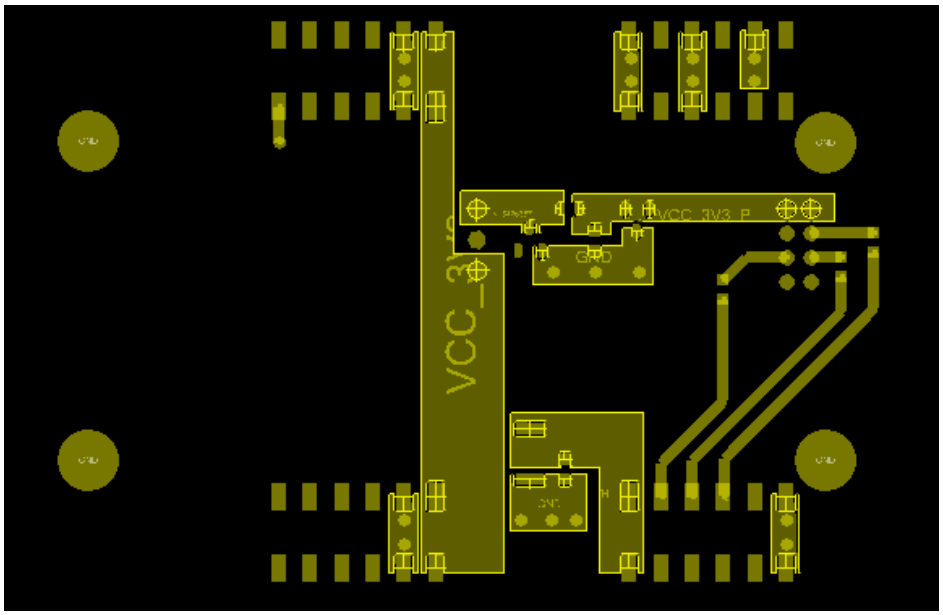
6.5 ANEXO V: CAPAS DE LA PLACA DE CIRCUITO IMPRESO

6.5.1 Capa conductora de cobre de la cara superior, *TOP*:



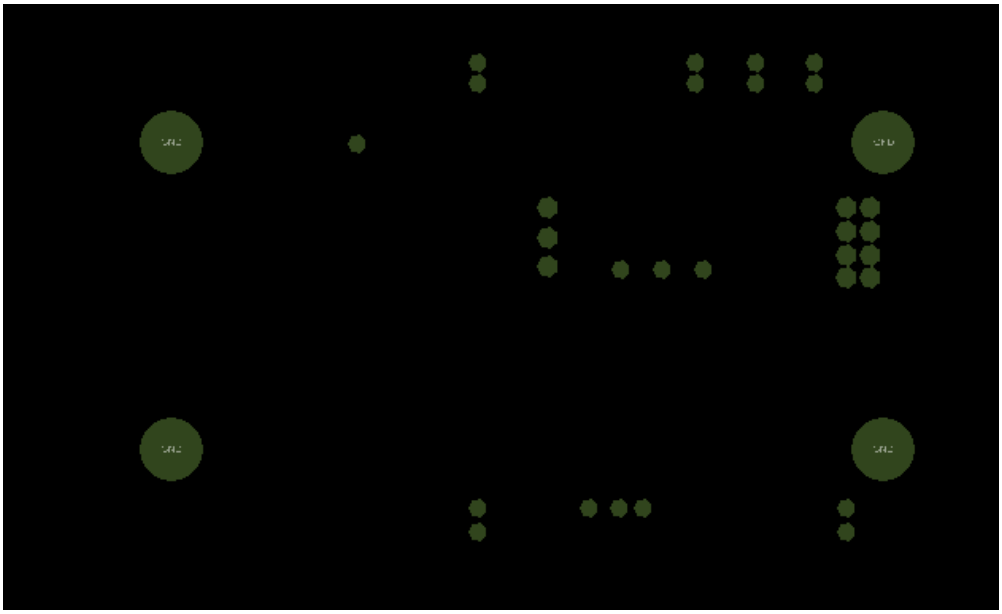
Captura 39. Capa conductora de cobre de la cara superior, TOP.

6.5.2 Capa conductora de cobre de la cara inferior, *BOTTOM*:



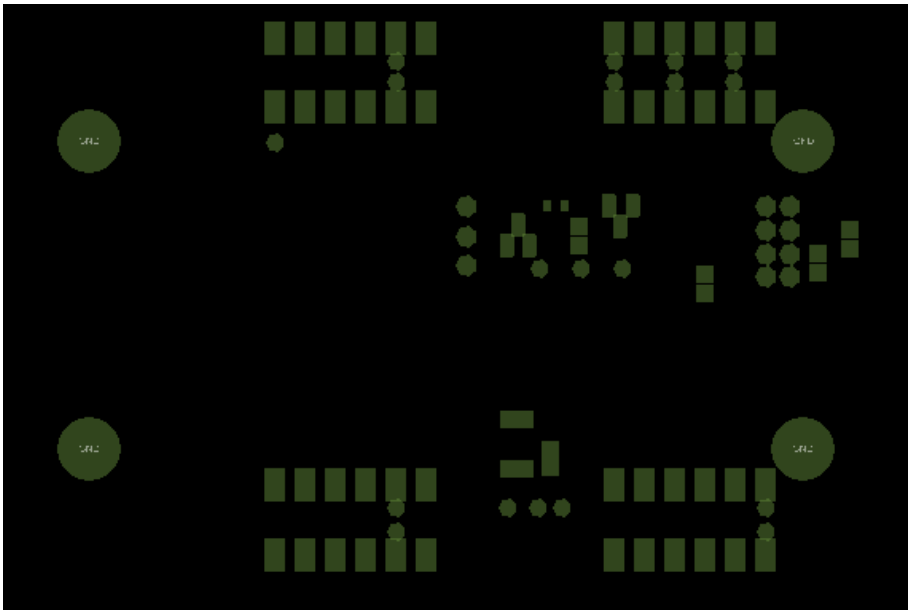
Captura 40. Capa conductora de cobre de la cara inferior, BOTTOM,

6.5.3 Capa de máscara de soldadura de la cara superior, *SMT*:



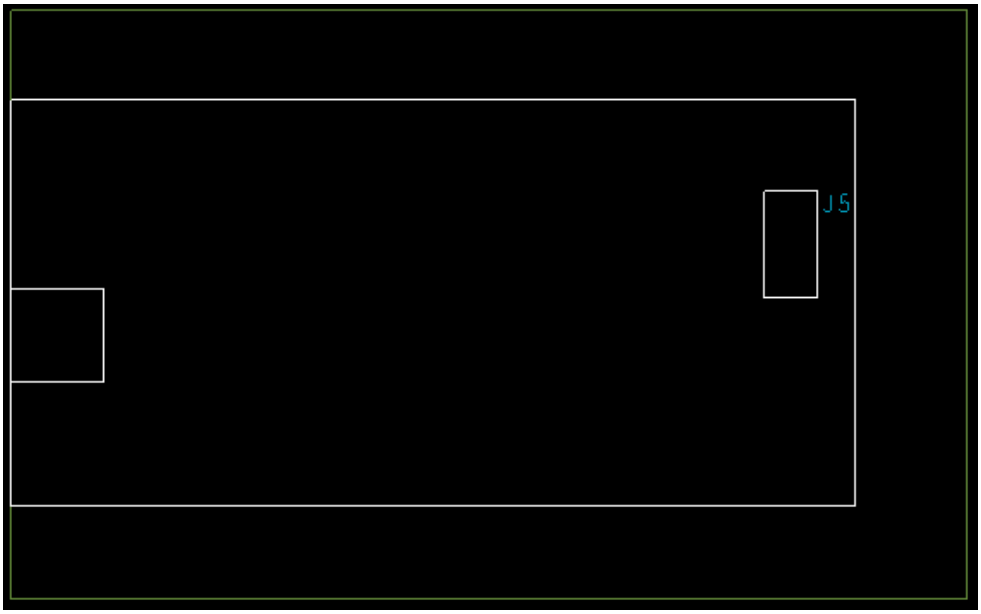
Captura 41. Capa de máscara de soldadura de la cara superior, SMT.

6.5.4 Capa de máscara de soldadura de la cara inferior, *SMB*:



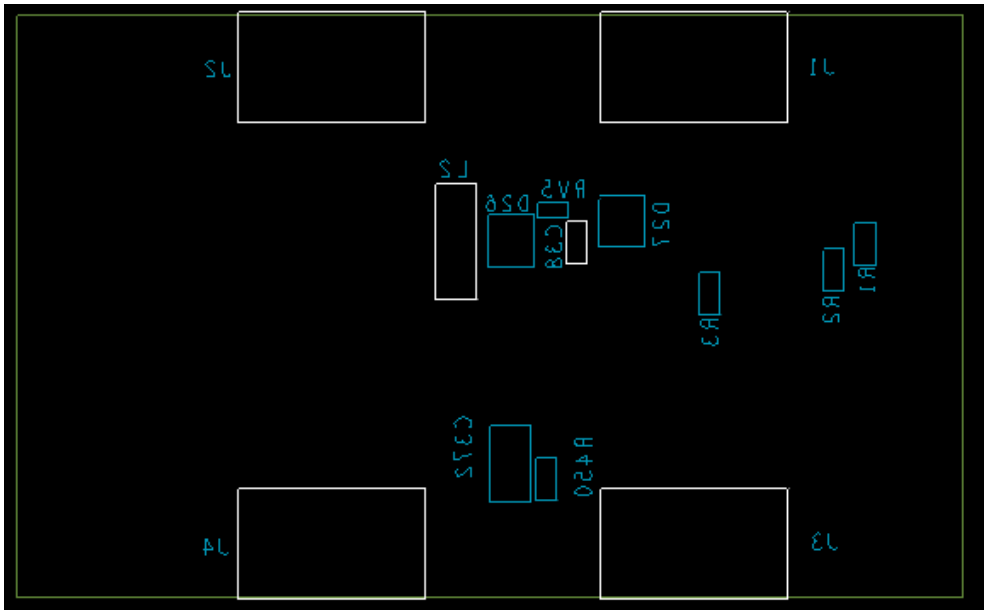
Captura 42. Capa de la máscara de soldadura de la cara inferior, SMB.

6.5.5 Capa de serigrafía de la cara superior, *SST*:



Captura 43. Capa de serigrafía de la cara superior, SST.

6.5.6 Capa de serigrafía de la cara inferior, *SSB*:



Captura 44. Capa de serigrafía de la cara inferior, SSB.

6.6 ANEXO VI: PROGRAMA CONFIGURACIÓN PUERTO SERIE DEL MÓDULO ICM STM 360

A continuación, se presenta el código utilizado en el ejecutable que cambia el protocolo de salida del puerto serie de *TSIP* a *NMEA*:

```
##Librerías
import time
import serial   ##Funciones para el tratamiento de la comunicación serie

##Serial Port Setup
ser = serial.Serial(
   port = '/dev/serial0',
   baudrate = 115200,
   parity = serial.PARITY_ODD,
   bytesize = serial.EIGHTBITS,
   stopbits = serial.STOPBITS_ONE,
   timeout = 1
)

##TSIP Package Structure Variables
p_dle = b'\x10'
p_etx = b'\x03'

##Set up NMEA
ser.write(p_dle)   #Start Packet
ser.write(b'\xBC') #0: p_id configure ports
ser.write(b'\x00') #1: Port to change
ser.write(b'\x0B') #2: Input baud rate
ser.write(b'\x07') #3: Output baud rate
ser.write(b'\x03') #4: Data bits
ser.write(b'\x01') #5: Parity
ser.write(b'\x00') #6: Stop bits
ser.write(b'\x00') #7: Reserved
ser.write(b'\x02') #8: Input protocol
ser.write(b'\x06') #9: Output protocol (NMEA)
ser.write(b'\x00') #10: Reserved
ser.write(p_dle)   #End Packet
ser.write(p_etx)

##WAIT
time.sleep(1)

##Se cierra el recurso
ser.close()
```

El recurso utilizado para realizar la transmisión hace uso del puerto serie y al ser la primera comunicación que recibe la tarjeta *ICM STM 360* es necesario que se atenga a la configuración por defecto del puerto. Esta configuración viene especifica en el manual de usuario.

Característica	*TXDA*	*RXDA*
Velocidad de transmisión	115200 bps	115200 bps
Bits de datos	8 bits	8 bits
Paridad	Impar	Impar
Bits de parada	1 bit	1 bit
Control de flujo	Ninguno	Ninguno
Protocolo	*TSIP*	*TSIP*

Tabla 11. Configuración por defecto de los puertos de entrada/salida del módulo ICM STM 360.

Por tanto, el recurso serie *"ser"* que se crea para enviar el mensaje TSIP tiene estas características, como se puede observar en el código del ejecutable. En el mensaje de configuración de puerto *0xBC* que se transmite no se alteran los valores del puerto de entrada porque no es necesario para la aplicación, pero con la siguiente tabla encontrada en el manual de usuario pueden modificarse.

Byte	Item	Type	Value/Unit	Description
0	Packet ID	UINT8	0xBC	
1	Port to Change	UINT8	0 1 0xFF	Port 1 (Standard) Port 2 (factory only) Current port
2	Input Baud Rate	UINT8	6 7 8 9 10 11	4800 baud 9600 baud 19200 baud 38400 baud 57600 baud 115200 baud
3	Output Baud Rate	UINT8	As above	As above
4	# Data bits	UINT8	2 3	7 bits 8 bits
5	Parity	UINT8	0 1 2	None Odd Even
6	# Stop bits	UINT8	0 1	1 bit 2 bits
7	Reserved	UINT8	0	
8	Input Protocols	UINT8	0 2 6	None *TSIP* *NMEA*
9	Output Protocols	UINT8	0 2 6	None *TSIP* *NMEA*
10	Reserved	UINT8	0	

Tabla 12. Valores del mensaje de configuración de puerto 0xBC.

6.7 ANEXO VII: CONFIGURACIÓN DE INTERVALO Y MASCARA DE PAQUETES DE LA TRANSMISIÓN DE PAQUETES NMEA

Si el puerto de salida se encuentra configurado para enviar el torrente de datos del GPS en formato NMEA, puede cambiarse tanto cada cuantos segundos se desea recibir la información como los paquetes que se transmiten por intervalo. Para ello se debe construir el siguiente mensaje como se especifica en el manual de usuario.

Byte	Item	Type	Value	Description
0	Packet ID	UINT8	*0x7A*	
1	Subpacket ID	UINT8	*0x00*	
2	Interval	UINT8	1-255	Fix Interval in seconds
3-6	Bit Mask Values (See below)			

Tabla 13. Valores del mensaje de configuración del protocolo NMEA.

Donde la máscara de bits tiene los siguientes valores hexadecimales que deben sumarse para generar los 4 bytes que deben enviarse junto con el resto del mensaje.

Message		Bit Mask
GGA	*GPS* fin data	0x00000001
GLL	Position fix, time of position fix and status	0x00000002
VTG	Course over ground and Ground speed	0x00000004
GSV	Satellites in view	0x00000008
GSA	*GPS DOP* and Active Satellites	0x00000010
ZDA	Time and Date	0x00000020
RMC	Recommended Minimum Specific *GPS* Data	0x00000080
GRS	*GNSS* Range Residuals	0x00000100
GBS	*GNSS* Satellite Fault Detection	0x00000200
GST	*GPS* Pseudo range Noise statistics	0x00000400

Tabla 14. Valores de la máscara de bits para determinar que paquetes envía el torrente de datos NMEA.

6.8 ANEXO VIII: CONFIGURACIÓN DEL FICHERO /BOOT/CONFIG.TXT:

Ejecutando el comando *"sudo nano /boot/config.txt"* se puede acceder al fichero para añadir las siguientes líneas.

```
#Habilitar el recurso ttyAMA0 para el GPSd
        dtparam=spi=on
        dtoverlay=pi3-miniuart-bt
        core_freq=250
        enable_uart=1
        force_turbo=1

#Habilitar el PPS para el GPSd
        dtoverlay=pps-gpio,gpiopin=4
```

De esta forma se consigue crear el recurso *pps0* en el pin de propósito general GPIO4 y el puerto serie utiliza el *ttyAMA0*.

6.9 ANEXO IX: CONFIGURACIÓN DEL FICHERO /ETC/DEFAULT/GPSD:

Ejecutando el comando *"sudo nano /etc/default/gpsd"* se puede acceder al fichero para añadir las siguientes líneas que configuran el daemon de *GPS*. A continuación, se presenta el contenido del archivo en su totalidad. Se observa cómo se especifican los recursos que necesita para sincronizarse: *ttyAMA0* y *pps0*.

```
# Default settings for the gpsd init script and the hotplug wrapper.

# Start the gpsd daemon automatically at boot time

        START_DAEMON="true"

# Use USB hotplugging to add new USB devices automatically to the daemon

        USBAUTO="false"

# Devices gpsd should collect to at boot time.

# They need to be read/writeable, either by user gpsd or the group dialout.

        DEVICES="/dev/ttyAMA0 /dev/pps0"

# Other options you want to pass to gpsd

        GPSD_OPTIONS="-n"

        GPSD_SOCKET="/var/run/gpsd.sock"
```