Taller de proyecto I

Facultad de ingeniería

Trabajo final

SITI: Sistema Integral de Transporte Inteligente

Integrantes

Cobanera Santiago

Iglesias Ramiro

Pontetto Axel

Ramallo Juan Manuel

20 de Febrero

2017

INDICE

1 INTRODUCCIÓN

Las extensas filas en las paradas de ómnibus, la falta de comodidad de los pasajeros durante el viaje, y la ausencia de información acerca de los horarios estipulados para cada ramal, son realidades cotidianas para todo aquel que utiliza el transporte público en la ciudad de La Plata.

Proveer a los usuarios información fidedigna sobre el transporte público, permite que puedan planificar mejor sus viajes, y permite que el transporte público pueda desarrollarse como alternativa real al uso del vehículo particular, ayudando a descongestionar la circulación en la ciudad.

Actualmente el sistema de transporte carece en su mayoría de una adecuada integración con las tecnologías modernas. El presente proyecto se basa en 3 principales funcionalidades que permiten contrarrestar esta tendencia

1. Localización Automática de Vehículos (AVL, por sus siglas en ingles)
2. Conteo Automático de Pasajeros (APC, por sus siglas en inglés)
3. Software para permitir una interacción productiva del usuario con la información de AVL y APC en tiempo real

El desarrollo de un prototipo que permita brindar las funcionalidades anteriormente descriptas, deberá cumplir una serie de requisitos que deben considerarse.

- Requisitos funcionales
    - Capacidad de brindar geolocalización
    - Capacidad de transmitir los datos recabados
    - Detección de la cantidad de pasajeros en el micro
    - Brindar alimentación a los módulos respetando sus corrientes y tensiones máximas y mínimas
- Requisitos no funcionales
    - Se deberá utilizar la placa de desarrollo EDU-CIAA
    - Se deberá desarrollar un “poncho” para la EDU-CIAA con el circuito necesario
    - El proyecto deberá terminarse antes de febrero de 2017
    - Las conexiones del prototipo deben ser firmes, considerando su aplicación a un colectivo

2 OBJETIVOS

Para el desarrollo del prototipo, se fijaron un conjunto de objetivos a cumplir teniendo en cuenta las restricciones de tiempo y de presupuesto.

Se clasificaron en objetivos primarios, es decir, aquellos que deben cumplirse para considerar el trabajo completado, y objetivos secundarios, es decir, mejoras o accesorios condicionados a los plazos, presupuestos, disponibilidad.

- Objetivos Primarios
    - Procesar información de sensores infrarrojos, que sean capaces de percibir el movimiento de cuerpos (pasajeros) en su campo de visión.
    - Sistema de posicionamiento global (GPS) que permita determinar la posición de del vehículo
    - Recolección de los datos aportados por los módulos antes mencionados, con su posterior transmisión a teléfonos móviles mediante un módulo GPRS.
    - Creación de poncho para la EDU-CIAA incorporando los módulos en cuestión

- Objetivos Secundarios:
    - Desarrollo de página web la cual mostrara los datos de los vehículos (posición + cantidad de pasajeros).
    - Incorporar un servidor externo para almacenamiento de datos: página web, usuarios, registros, etc.
    - Desarrollo de aplicación para teléfonos móviles: Android y IOS.

En el presente informe se detalla de acuerdo al formato sugerido por la cátedra el proceso de desarrollo del prototipo. El código completo en lenguaje C, esquemáticos, PCB, y documentación necesaria, si bien se irán referenciando a lo largo del presente informe, pueden consultarse en el siguiente repositorio: https://github.com/juanmanuelramallo/SITI

3 DESARROLLO DE HARDWARE

3.1 DIAGRAMA EN BLOQUES

En la figura 3-1 se puede observar el diagrama en bloques del sistema SITI. Componiéndose el mismo por las dos principales fuentes de información: el GPS y los sensores, a su vez del módulo encargado de la comunicación y transferencia de datos: el módem GPRS, Junto a la unidad central del procesamiento del sistema: la CIAA. Si bien los satélites no forman parte de nuestro alcance de trabajo, son necesarios para la utilización del GPS. El servidor al final de la comunicación es quien se encarga del almacenaje de la información recolectada, para luego poder consumir tales datos ya sea en un página web o aplicación móvil.

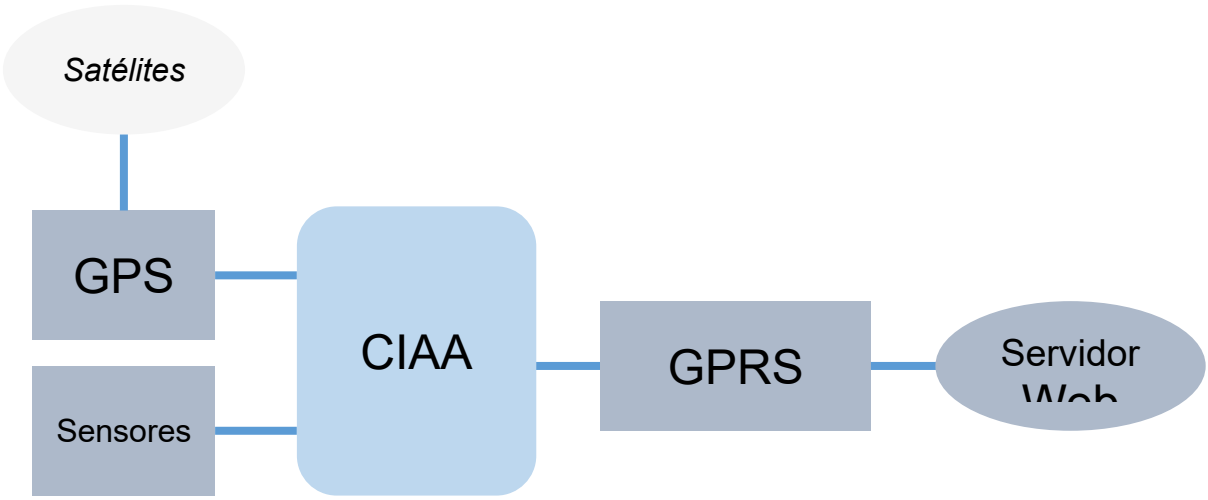


Figura 3-1

3.2 INTERFACES ELÉCTRICAS PONCHO

En las figuras 3-2 y 3-3 podemos observar la disposición de las interfaces eléctricas de ambas caras. Luego se expondrá las correspondencias de cada una de las interfaces correspondientes a cada módulo por separado.

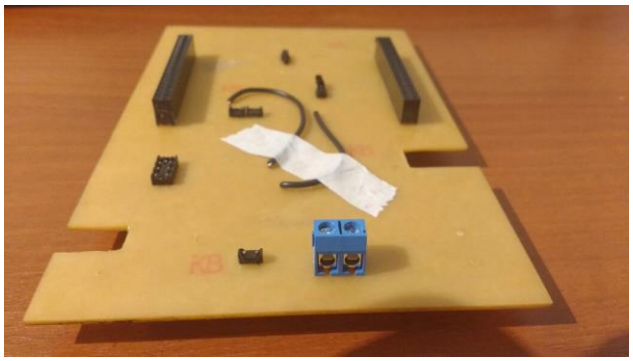


Figura 3-2

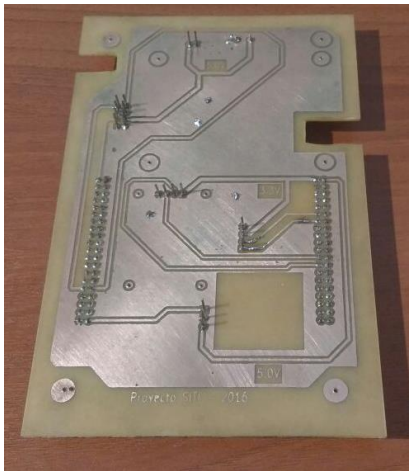


Figura 3-3

3.3 INTERFAZ EXTERNA DE ALIMENTACIÓN DEL PONCHO

El poncho dispone de una interfaz de alimentación externa de 5 volts para alimentar a la EDU-CIAA y al módulo de comunicación GPRS. La misma puede observarse en la figura 3-4.

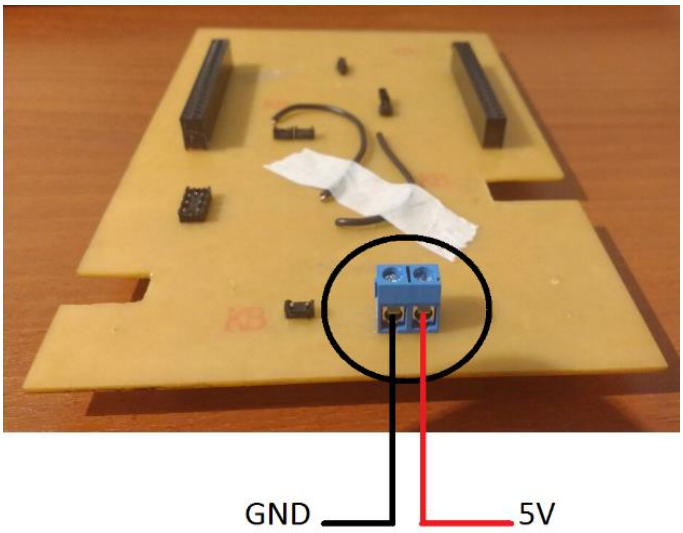


Figura 3-4

3.4 INTERFAZ DE ALIMENTACIÓN DE LA EDU-CIAA

El poncho dispone de un conector de alimentación derivado de la alimentación externa de 5 volts para alimentar específicamente a la EDU-CIAA. El mismo puede observarse en la siguiente figura (Figura 3-5).

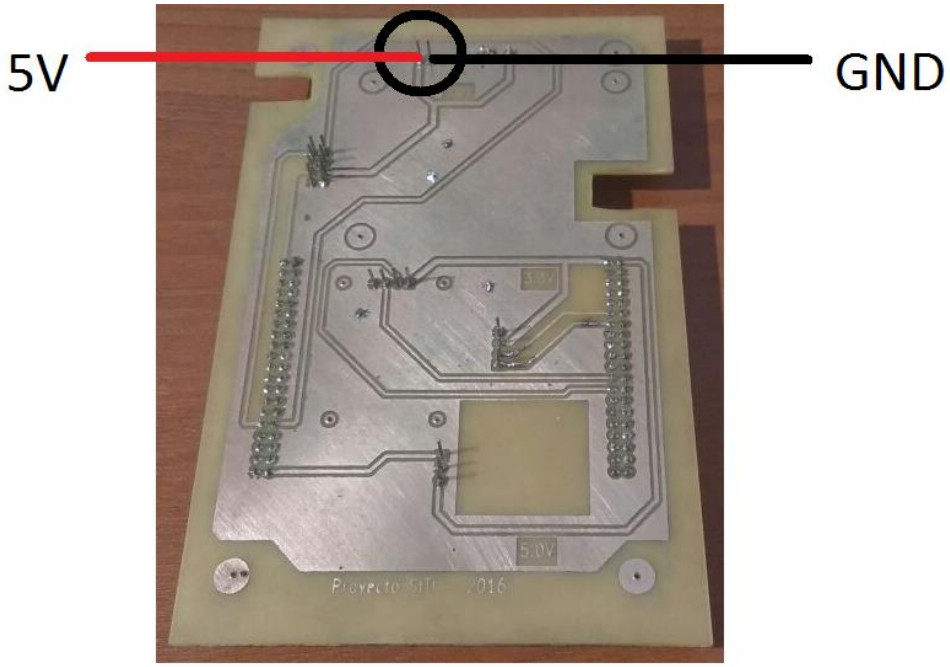


Figura 3-5

3.5 INTERFAZ ELÉCTRICA DEL MÓDULO GPS

El módulo de GPS posee su propio conector sobre el poncho, con 4 pines como puede observarse en la figura 3-6.

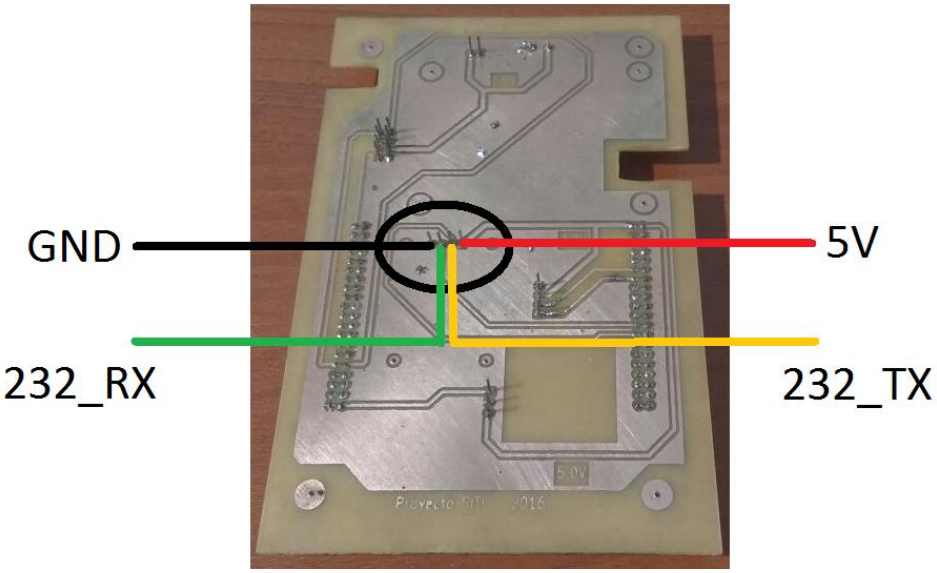


Figura 3-6

3.6 INTERFAZ DE ALIMENTACIÓN DEL MÓDULO GPRS

Al igual que el modulo GPS, el módulo de GPRS posee su propio conector sobre el poncho, el cual puede observarse en la figura 3-7. Como se puede apreciar hay dos líneas de alimentación, por un lado, la línea de alimentación de 5V es usada para energizar el circuito del módulo GPRS, debido a que este demanda una corriente de 2A, la cual es provista por la alimentación externa. Por otro lado, se tiene una línea de alimentación de 3.3V, la cual es provista por la EDU-CIAA, y se utiliza como valor de referencia para la representación de los valores lógicos con los cuales trabaja dicho módulo.

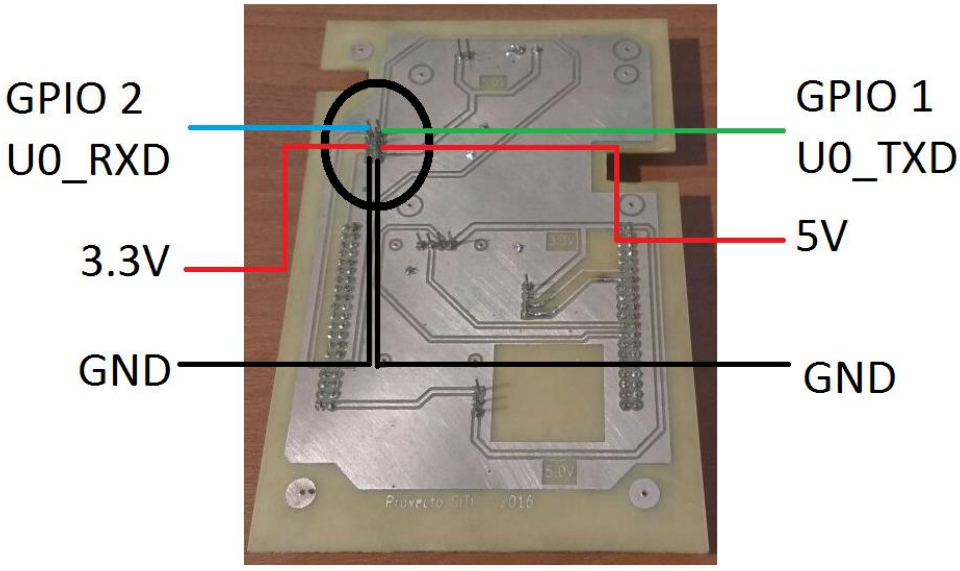


Figura 3-7

3.7 INTERFAZ DE CONEXIÓN PONCHO EDU-CIAA

En la figura 3-8 podemos observar los bancos de pines que brindan el soporte de conexionado entre el poncho y la EDU-CIAA.

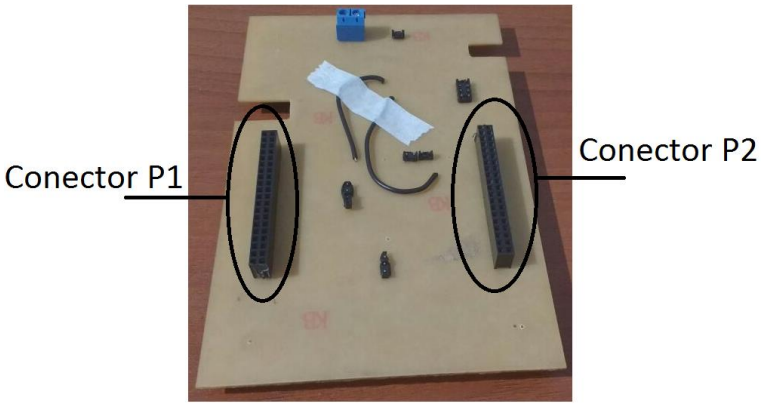


Figura 3-8

3.8 INTERFAZ DE USUARIO

Dado que el módulo SITI está orientado a ser usado para recolectar y transferir información a la nube, su interfaz de usuario (de parte del hardware) se puede describir de la siguiente manera. Las interacciones disponibles para que el usuario común pueda ejecutar son:

- Con un botón, encender el módem GPRS
- Con un botón, encender el sistema completo de SITI (enciende la CIAA)
- El GPS no requiere interacción
- Los sensores no requieren interacción

Se puede observar en la figura 3-9 una idea sobre la distribución de ambos botones en el módulo central de SITI.

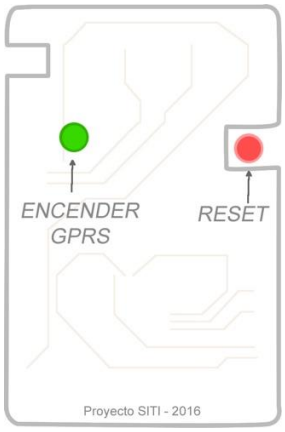


Figura 3-9

Como a su vez el sistema SITI proveerá un medio para distribuir la información recolectada, los cuales son la página web y las aplicaciones móviles, estos mismos tendrán su propia interfaz de usuario y manejo propio de tales subsistemas dependientes totalmente de SITI. Dado que son objetivos secundarios, las interfaces de usuario de tales aplicaciones no se describen en este documento.

3.9 COMPONENTES Y CIRCUITOS

3.9.1 Sensores Infrarrojos

Tanto los receptores como emisores infrarrojos se muestran a continuación en el siguiente el esquemático con su respectiva conexión a la CIAA, ver figura 3-10.

Como se logra apreciar en la figura 3-11 los emisores son alimentados en series con una tensión de 5V y comandados por un transistor 2n222 el cual controla el encendido de los led (conectado al GPIO8). Por otro lado, los receptores son alimentados con una tensión 3,3 V, la cual es recibida en canal del ADC en caso de el fototransistor reciba luz infrarroja.

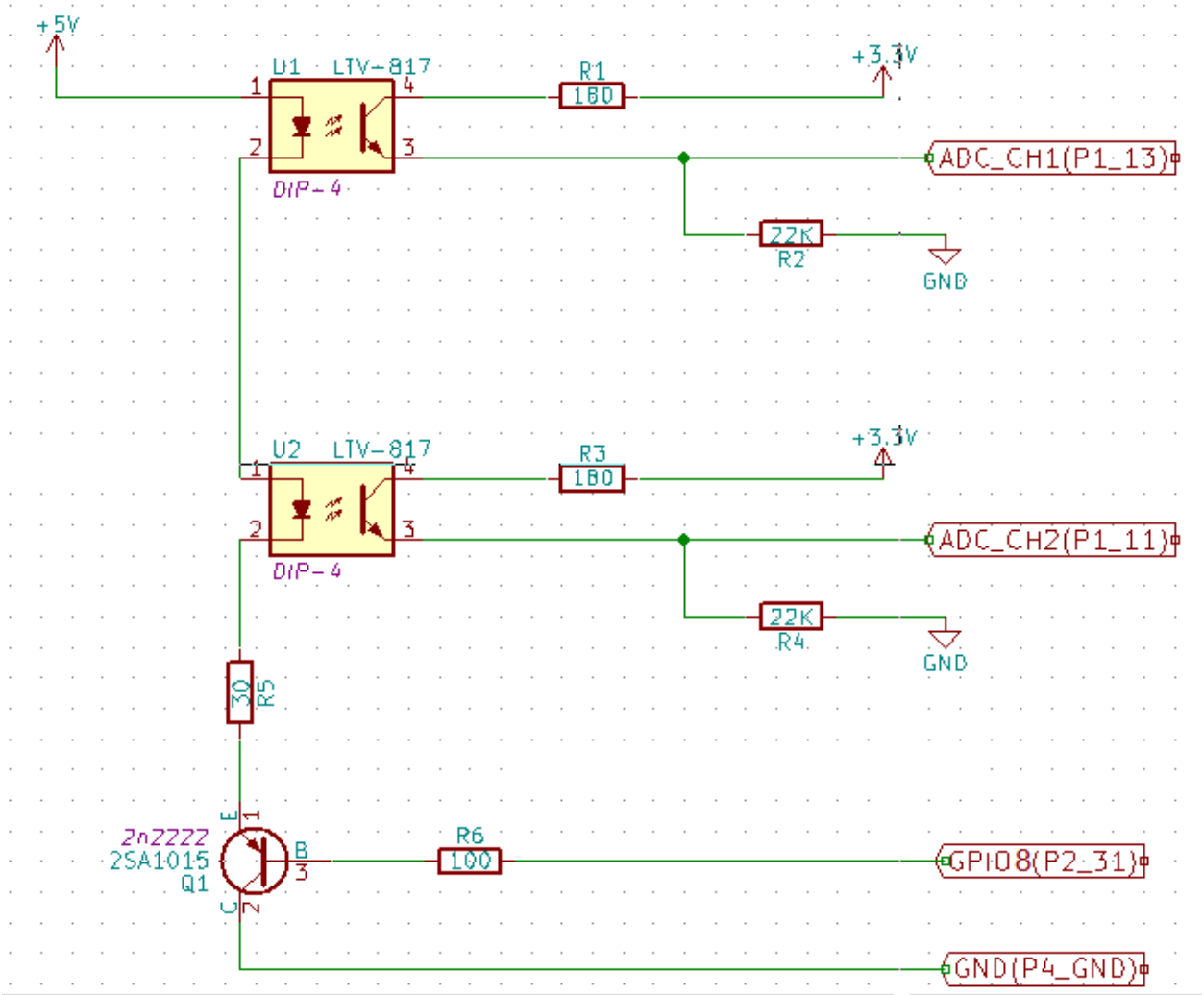


Figura 3-10

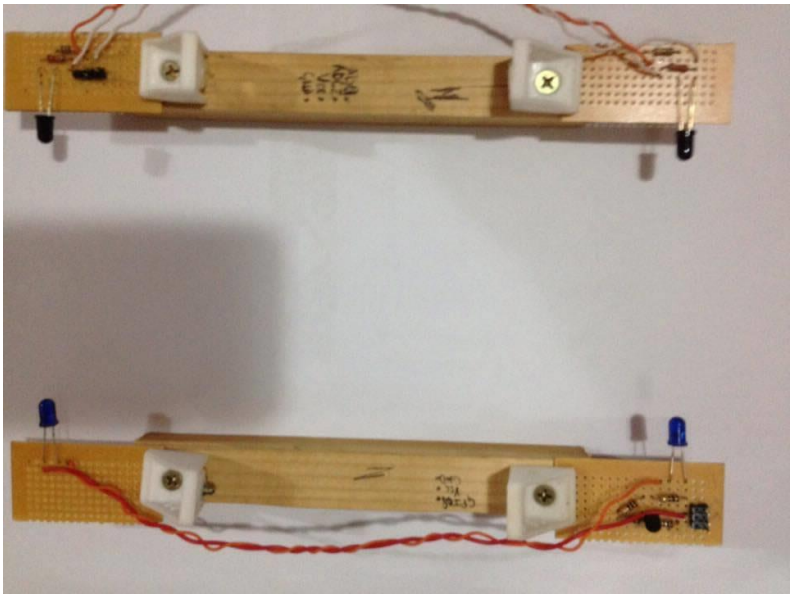


Figura 3-11

Los componentes del contador de pasajeros compuesto por los sensores infrarrojos son:

- 2 Emisores infrarrojos (IR333-A)
- 2 Receptores infrarrojos (PD333-3B/H0/L2)
- 2 resistencias de 180 Ω
- 2 resistencias de 22k Ω
- 1 resistencia de 100 Ω
- 1 resistencia de 30 Ω

El detalle de alimentación para los sensores y los emisores infrarrojos se detalla en la siguiente tabla:

Sensores infrarrojos	
Tensión	3,3 V
Corriente	5 mA
Emisores infrarrojos	
Tensión	5V
Corriente	120 mA

Tabla 3-1

3.9.2 Módulo GPS

La interface del GPS (GY-GPS6MV2) nos permite protección al módulo GPS (U-blox NEO-6M GPS module) y acceso a los pines de comunicación serie Rx y Tx.

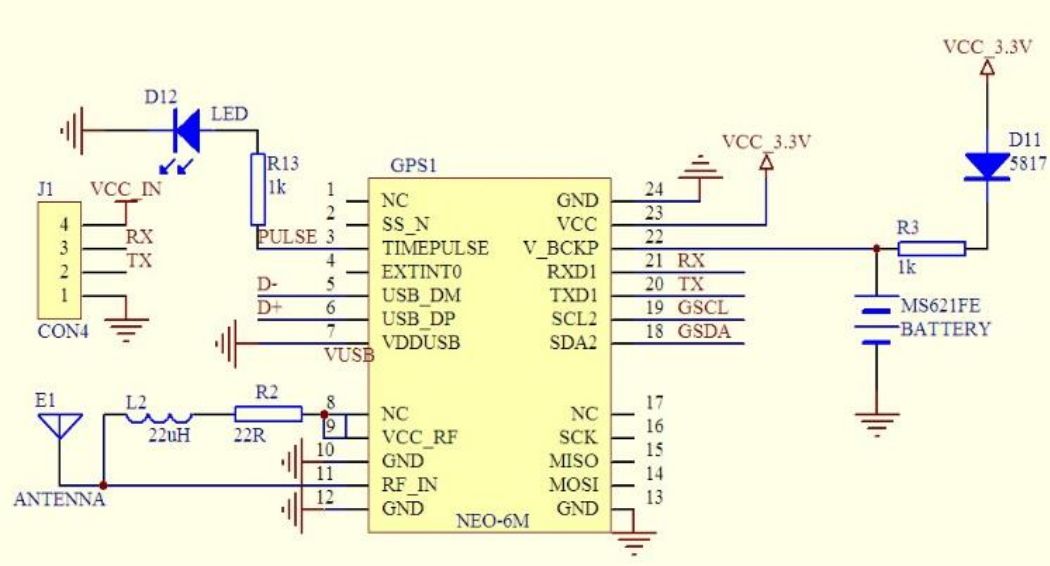


Figura 3-12

En la Figura 3-13 se muestra una imagen ilustrativa de la interface. Para el conexionado a la CIAA ver la figura 3-14.

Figura 3-13

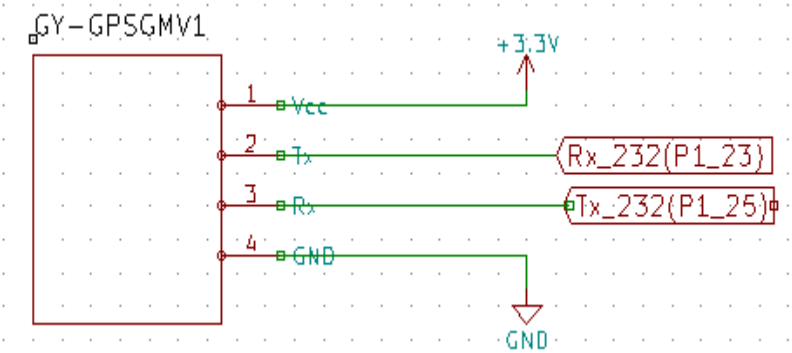


Figura 3-14

La alimentación del GY-GPSGMV1 está dada por la siguiente tabla, ver tabla 3-2.

GPS	
Tensión	3,3 V
Corriente	100 mA

Tabla 3-2

3.9.3 Módulo GPRS

A continuación, se ilustra un diagrama de la interface AN0002_SIM808 la cual utiliza un módulo SIM808 GSM/GPRS/GPS, ver figura 3-15.

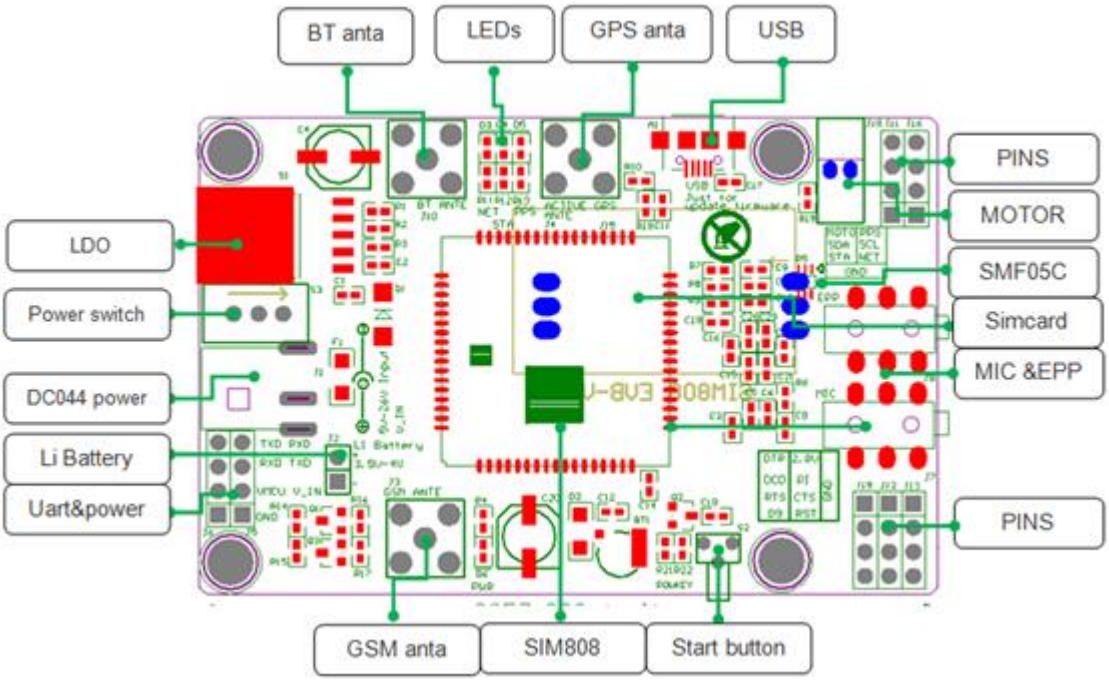


Figura 3-15

En la Figura 3-16 se muestra una imagen ilustrativa de la interface. Para el conexionado a la CIAA ver la figura 3-17.

Figura 3-16

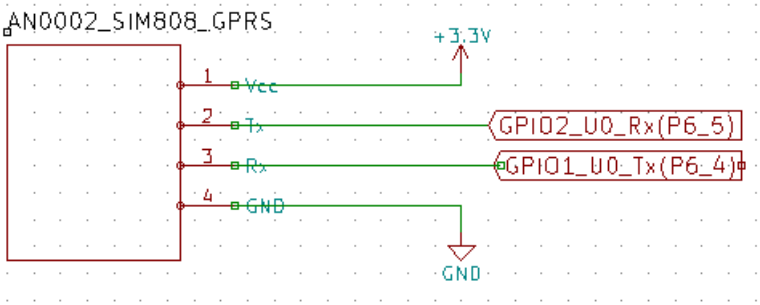


Figura 3-17

La interfaz de alimentación DC044 del AN0002_SIM808 responda a los siguientes valores (ver *SIM808 SHEILD V1.0 User Manual*):

GPRS	
Tensión	5 – 26 V
Corriente	2 A

Tabla 3-3

3.10 Diseño PCB

Para el diseño del PCB debieron analizarse las dimensiones y conexionados de los distintos módulos utilizados en el proyecto.

Como primera decisión de diseño, debido a las dimensiones de los dispositivos a montar sobre el poncho, se optó por el poncho grande que cubre toda la EDU-CIAA. Se dejaron huecos para el pulsador de reset, y la alimentación. En función de la interfaz de usuario que se definió no se consideró necesario dejar huecos para los pulsadores y los LEDs.

Resulta de suma importancia analizar la alimentación que se utilizará. El resumen de consumo para cada componente del sistema es el siguiente:

Módulo	Tensión [V]	Consumo MAX [mA]
EDU-CIAA	5,0	300
GPS	3,3	100
GPRS	5,0	2.000
Emisores IR	5,0	130
Receptores IR	3,3	20

Tabla 3-4

Nota: el detalle de la tolerancia para cada uno de los dispositivos fue considerado, y se puede consultar en las hojas de datos que se referencian en la bibliografía

Considerando lo anteriormente detallado, se observa que el consumo total será, como máximo, de 2,55 Ampere. En base a lo expuesto, se establece que el prototipo requiere una alimentación externa de 5,0V con una tolerancia de 5%, y capacidad de entregar 2,55A.

Se utiliza un conector robusto en el poncho, donde se conectará la fuente de alimentación externa. A partir de allí, se diseña un circuito capaz de proveer la alimentación necesaria a cada módulo.

Como muestra la figura 3-18, la EDU-CIAA y el módulo GPRS se alimentarán directamente de la fuente externa. Por otra parte, el GPS y los sensores (cuyo consumo total nunca superará los 250mA), serán alimentados desde los pines de la EDU-CIAA.

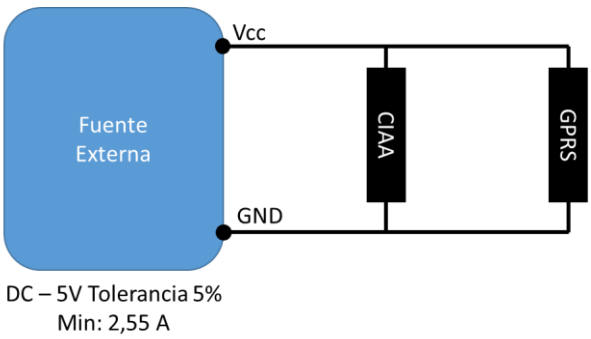


Figura 3-18

Por lo tanto, los componentes que deben considerarse en el poncho incluyen:

1- Conector de 3 pines para emisores infrarrojos
2- Conector de 4 Pines para receptores infrarrojos
3- Módulo GPS (25mm x 35mm)
4- Módulo GPRS (50mm x 76mm)
5- Conector para alimentación externa
6- Conector para alimentar a la EDU-CIAA

La distribución utilizada para los distintos componentes se representa esquemáticamente en la siguiente figura, ver figura 3-19.

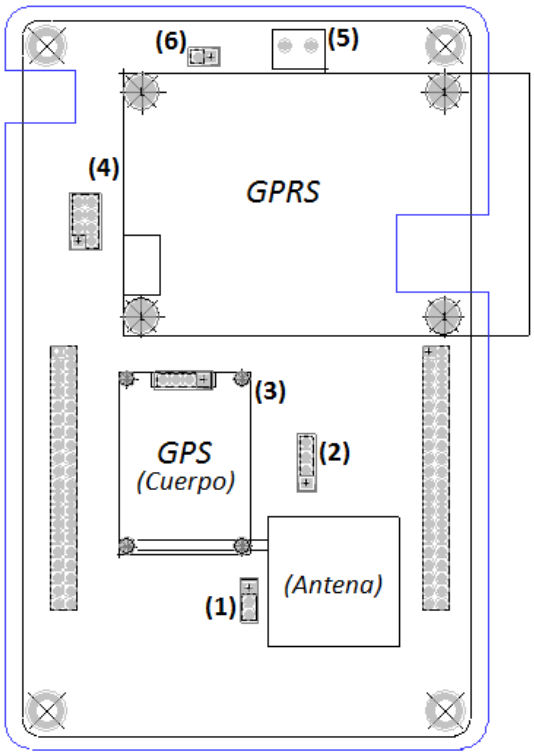


Figura 3-19

Se realizó un cuidadoso análisis de las pistas para obtener un recorrido óptimo, incorporando indicaciones a las pistas de alimentación para evitar errores de conexiones que pudieran dañar algún componente. En el Anexo 1 puede observarse el resultado del diseño del PCB.

4 DESARROLLO DE SOFTWARE

4.1 GPRS

4.4.1 Descripción

La gran mayoría de los módems GPRS actuales utilizados en IoT son controlados por medio de comandos AT (ver anexo 2). Entre las interfaces de comunicación disponibles (UART y IIC) optamos por usar la UART aprovechando las múltiples interfaces del tipo disponibles en la CIAA.

La idea principal de cómo transferir datos a un servidor web consiste en ejecutar un pedido http, más específicamente un verbo http, tal verbo es llamado POST. Se debe hacer un POST dado que así lo especifican los documentos del protocolo cuando se trata de subir información.

Tal información se almacena en un servidor web externo, tal servidor externo es una plataforma como servicio llamada Ubidots (www.ubidots.com). Esa plataforma se encarga de crear *endpoints* o rutas de acceso para agregar información a ciertas variables que hayamos definido en el servicio. A su vez nos provee la facilidad de leer los datos subidos a través de una API que luego será usada por la página web y aplicaciones móviles. Para tener una idea global del flujo de información en la comunicación de SITI se puede observar la figura 4-1 para un detalle simplificado el proceso.

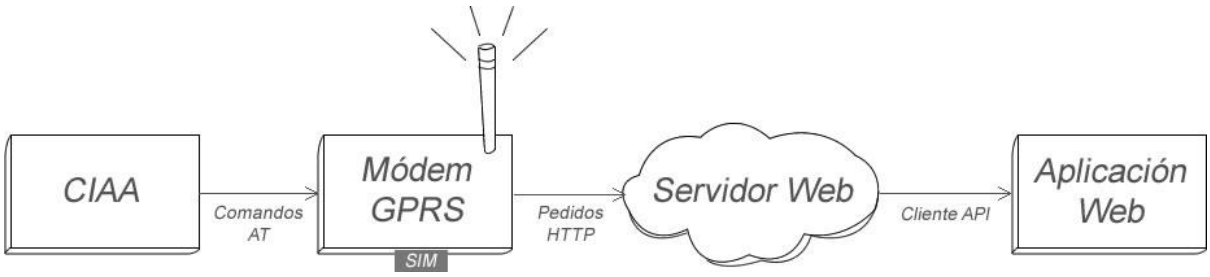


Figura 4-1

Para la resolución de la comunicación se creó una librería con las funciones requeridas para que el planificador ejecutara. Se consideró hacer cada función de manera que no bloqueara la ejecución continua el bucle principal, dado que el contador de pasajeros no puede permitirse tener demora en su atención por parte del procesador. A continuación, se detallan las partes más importantes de la solución.

4.1.2 Diagrama de estados

Fue fundamental el hecho de armar una máquina de estados para la ejecución normal del módem GPRS dado que de otra forma las funciones habrían sido bloqueantes. Es por eso que se implementó la máquina con los siguientes estados:

- INIT: estado en el cual se encuentra al inicializar el módem
- SENDING: estado en el cual se encuentra al efectuarse un envío. Tiene lo sub-estados siguientes:
    - STRING: se envía el string preparado vía UART hacia el módem GPRS
    - ACTION: se envía el comando para efectuar el verbo http
- READING: usado para debugging
- IDLE: listo para ejecutar alguna tarea y sin tareas pendientes

En la figura 4-2 se puede apreciar claramente la transición entre un estado y otro.

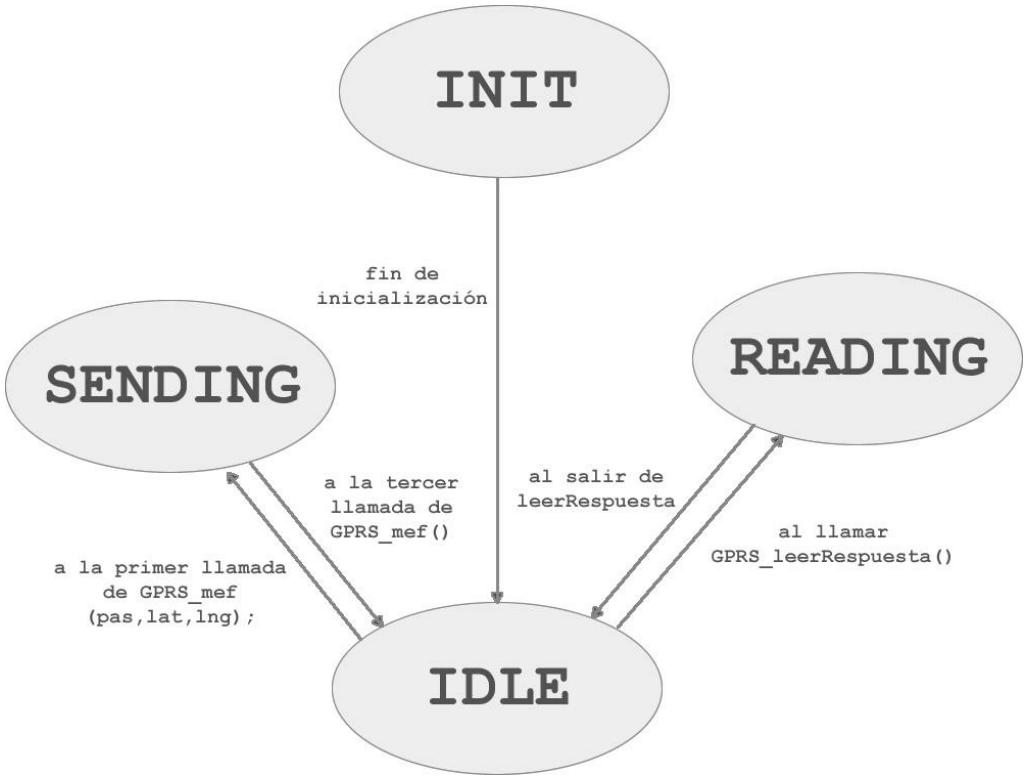


Figura 4-2

4.1.3 Comandos AT utilizados

A lo largo del proyecto se han usado distintos comandos AT hasta encontrar los cuales mejor satisfacían nuestras necesidades. Entre los cuales se encuentran lo siguientes con un pequeño detalle de su uso:

- AT: retorna OK si el módem está listo para ser usado
- AT+CSQ: retorna la calidad de la señal en términos de dos parámetros, el received signal strength indication y el bit error rate.
- AT+SAPBR=<tipo de comando>,<id de contexto>: se usa para configurar los parámetros de red tanto como el APN, el usuario y la contraseña y así mismo para activar el estado de GPRS

- AT+HTTPINIT: inicializa el servicio http
- AT+HTTPPARA: configura los parámetros para el servicio http, tales como el url, el tipo de contenido, el id del contexto
- AT+HTTPDATA=<tamaño>,<tiempo>: sirve para ingresar los datos a enviar, se especifica el tamaño a enviar en bytes y el tiempo máximo para introducir los datos en milisegundos.
- AT+HTTPACTION=<tipo de acción>: sirve para efectuar la acción especificada, ya sea HEAD, POST o GET. Retorna la acción, un código de estado y la longitud de datos.

4.1.4 Pseudocódigos

Para la implementación de la MEF se planteó la siguiente propuesta de pseudocódigo:

```
Si el gprs está libre, (es decir estado actual en IDLE)
      Limpiar el string de respuesta
      Limpiar el string a enviar
      Armar el string a enviar con los parámetros
      Enviar el comando AT+HTTPDATA=50,2000
      Actualizar el estado actual a ENVIANDO
      Actualizar el subestado de ENVIANDO en STRING
Si no, si el gprs se encuentra enviando
      Si el subestado de ENVIANDO indica STRING
            Enviar el STRING armado previamente al módem GPRS
            Actualizar el subestado de ENVIANDO en ACTION
      Si no, si el subestado de ENVIANDO indica ACTION
            Enviar el comando AT+HTTPACTION=1
            Actualizar el estado actual a IDLE
```

Para la inicialización del GPRS se tiene el siguiente pseudocódigo:

```
Poner estado actual en INIT
Armar strings globales auxiliares
Limpiar string de respuesta
Probar comando AT
Pedir la calidad de la señal
Configurar los parámetros del GPRS
      APN, user, password
Activar el estado GPRS
Mostrar IP asignada por ISP
Inicializar el servicio http
Configurar parámetros para http
      Context id, URL, content
Actualizar estado en IDLE
```

Para saber lo que el módem GPRS respondía a nuestros pedidos se implementó la función leerRespuesta() que funciona de la siguiente manera:

```
Escribir en el UART_USB el string de respuesta
Limpiar el string de respuesta
```

Pero vale mencionar que el string de respuesta es una variable global en el módulo de software gprs.c y que es usado con el modificador extern en el archivo sAPI_UART.c donde se definen las interfaces UART a usarse. El manejador de la interrupción para la UART asignada al GPRS toma el carácter enviado desde el módem y lo agrega al final de un string que luego será leído por el método leerRespuesta() mencionado previamente.

4.2 GPS

4.2.1 Descripcion

Para la geolocalizacion de las diferentes unidades de transporte, colectivos particularmente, se utiliza un dispositivo de gps (sistema de posicionamiento global) conectado a la EDU-CIAA-NXP, de modo de permitir al usuario disponer de las coordenas de dichas unidades de tranportes en todo momento.

El dispositivo usado para tal fin es el GY-GPS6MV2, el cual cuenta con un chip integrado Ublox NEO-6M como se explicó en la sección *Desarrollo de Hardware*

4.2.2 NMEA 0183

Antes de comenzar con las implicancias específicas de software relacionadas al módulo gps, es conveniente explicar el protocolo que utilizan los mismos para presentar los datos, denominado protocolo NMEA 0183.

NMEA 0183 es una especificación combinada eléctrica y de datos entre aparatos electrónicos marinos y, también, más generalmente, receptores GPS. El protocolo NMEA 0183 es un medio a través del cual los instrumentos marítimos y también la mayoría de los receptores GPS pueden comunicarse los unos con los otros. Ha sido definido, y está controlado, por la organización estadounidense National Marine Electronics Association.

Este protocolo presenta los datos en forma de tramas, las cuales pueden ser de diferentes tipos, como ser: GPGGA, GPGSA, GPRMC, etc. Por un lado, si bien, encontramos amplia variedad de tipos de tramas, la mayoría de ellas nos brindan la información como la hora y zona horaria, latitud, longitud y dirección. Por otro lado, uno de los tipos de tramas mas utilizados es GPGGA, motivo por el cual decidimos incluirla en el proyecto, sumado a que nos brinda datos adicionales que serán de gran interés a la hora de escalar el proyecto, por lo que, se procede a explicar la trama NMEA 0183 GPGGA.

La trama NMEA 0183 GPGGA presenta la siguiente estructura de mensaje genérico:

*$GPGGA,hhmmss.ss,Latitude,N,Longitude,E,FS,NoSV,HDOP,msl,m,Altref,m,DiffAge,DiffStation*cs<CR><LF>*

donde cada uno de los campos correspondientes se observa en la siguiente tabla (tabla 4-1):

CAMPO	FORMATO	UNIDAD	DESCRIPCION
$	Carácter		Indicador de comienzo de trama (mensaje)
$GPGGA	String		Identificador del mensaje, protocolo GGA (global positioning system fix data)
hhmmss.ss	hhmmss.ss		Tiempo actual en formato UTC
Latitude	ddmm.mmmm		Latitud, grados + minutos
N	Caracter		Indicador N/S, N: norte o S: sur
Longitude	dddmm.mmmm		Longitud, grados + minutos
E	Caracter		Indicador E/O, E: este u O: oeste
FS	Digito		Indicador de estado de la posición (indicador de la calidad)
NoSV	Numérico		Satélites usados, rango: 0 – 12
HDOP	Numérico		Dilución Horizontal de Precisión
msl	Numérico	m	Altitud MSL (mean sea level – metros sobre el nivel del mar)
m	Caracter		Unidad de msl (m, metros)
Altref	Numerico	m	Separación de geoides
M	Carcter		Unidad de Altref (m, metros)
DiffAge	Numerico	s	Edad de los datos GPS diferenciales
DiffStation	Numérico		Identificación de estación de referencia diferencial
*cs	Hexadecimal		Suma de comprobación, comienza luego del carácter *
<CR><LF>	Caracter		Retorno de carro y nueva línea, indicador de final de la trama (mensaje)
,	Carácter		Separador de campos de la trama

Tabla 4-1

4.2.3 Diagramas de estados

En este apartado se muestran los diagramas de estados que gobiernan el control de los datos producidos por el GPS. Dicho diagrama de estados podemos dividirlo en dos partes, por un lado, el diagrama de la maquina de estados que muestra el parseo de cada uno de los datos (carácter por caracter especificamente) obtenidos por el gps y por otro lado, el diagrama que muestra como se controla dicho parseo y la presentación de los datos completos en pantalla para la verificación de ls mismos.

4.2.3.1 Diagrama de estados para el parseo de los datos

En la figura 4-4 podemos observar el diagrama de estados correspondientes al corazón de la lógica para el tratamiento de los datos brindados mediante el protocolo NMEA 0183 con trama GPGGA. Es importante aclarar que para los objetivos del proyecto basta con procesar solamente la información correspondiente al tipo de trama (GPGGA en nuestro caso), latitud, longitud y dirección. Dicho diagrama de estados representa la lógica del procedimiento de parseo (gpsFrameParse) cuyo seudocódigo será expuesto en su apartado correspondiente.

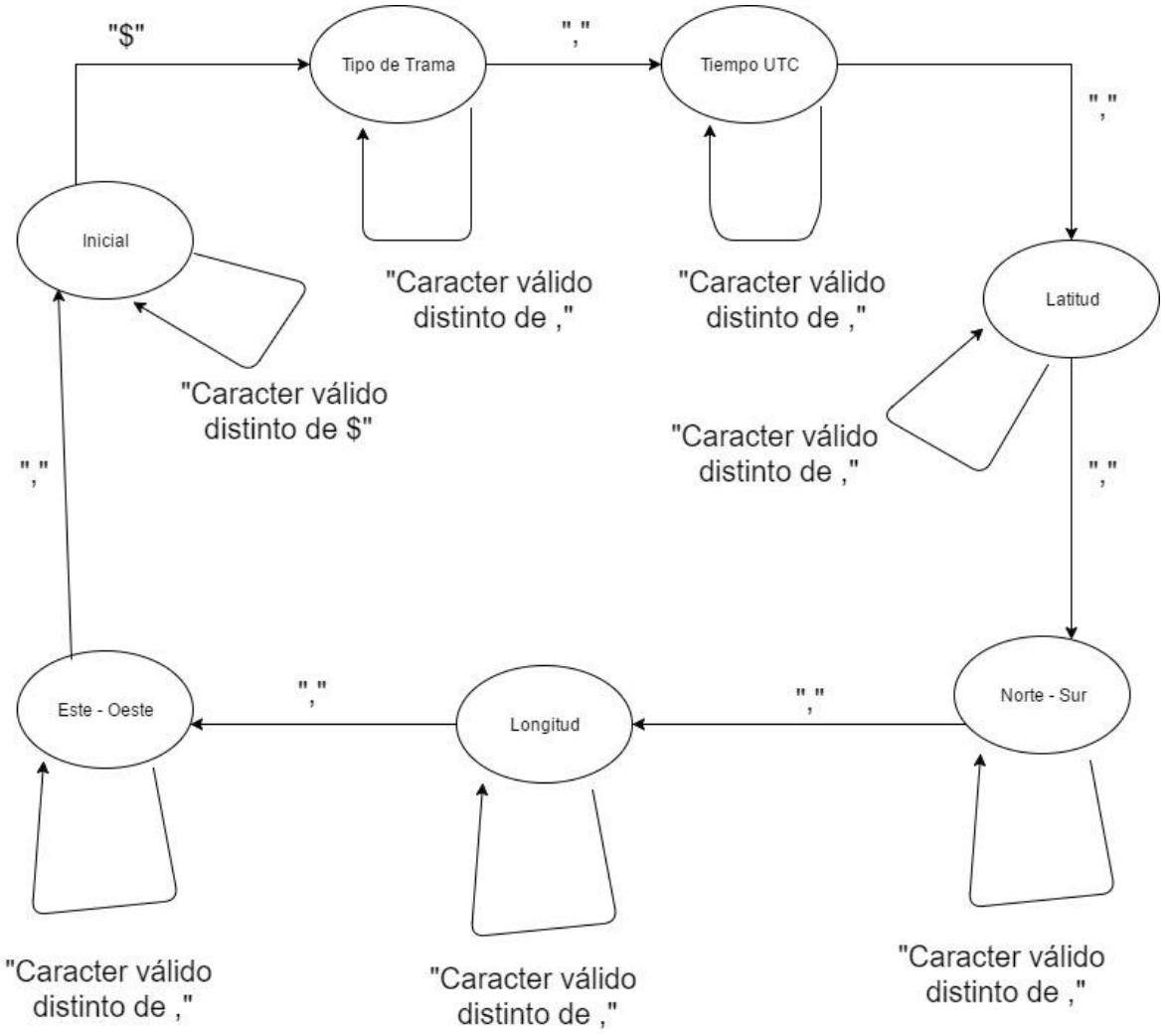


Figura 4-3

El funcionamiento de la máquina de estados mencionada es el siguiente: la entrada de datos está representada por un carácter, luego se tiene un estado inicial en donde se aguarda por la llegada del carácter indicador de comienzo de trama “$”, el cual motiva el cambio de estado al segundo estado denominado tipo de trama. En dicho estado se procesan cada uno de los caracteres que sean válidos mientras no se reciba el carácter delimitador de campos “,”, el cual, produce un nuevo cambio de estado hacia el estado denominado tiempo UTC, el cual representa el tiempo (horas minutos y segundos) en formato UTC y se procede de forma análoga para cada uno de los estados (datos a obtener) de interés: latitud, Norte-Sur, longitud y Este-Oeste.

Una vez finalizado el ciclo, es decir, concluido el estado Este-Oeste se activa un aviso (flag de indicación) de que se encuentran disponibles datos nuevos para procesar y se regresa al estado inicial para comenzar el tratamiento de una mueva trama.

Es importante aclarar que cualquier carácter invalido que se reciba hace que la máquina de estados regrese a su estado inicial, debido a que dicho carácter corrompe la trama en proceso. Esta situación se ha obviado en el diagrama para simplicidad y claridad del mismo pero debe tenerse presente.

Un carácter considerado valido para el procesamiento de la trama mencionada esta representado por los siguientes rangos de dominios de caracteres:

- a – z
- A – Z
- 0 – 9
- .
- ,
- $
- *
- \n
- \r

4.2.3.2 Diagrama de estados para el control del parseo y presentación de la información

Para analizar el diagrama de estados de la figura 4-5, se debe tener un esquema mental del funcionamiento básico del módulo GPS, el cual es el siguiente: cada vez que el modulo GPS obtiene una componente de un dato (carácter explícitamente), produce una interrupción a la EDU-CIAA, la cual atiende dicha interrupción y avisa (flag) que se ha recibido un dato del GPS. Dicho aviso es tomado por el planificador (el cual sera explicado en detalle más adelante) pasando el dato obtenido a la máquina de estado que se encarga del parseo de los datos, que ha sido presentada en el apartado anterior. Luego de que el parseo ha concluido con éxito, indicando que se han obtenidos los datos completos (coordenadas particularmente), situación que es detectada por el planificador, mediante el aviso (flag) correspondiente, se procede a visualizar los mismo en pantalla a modo de test.

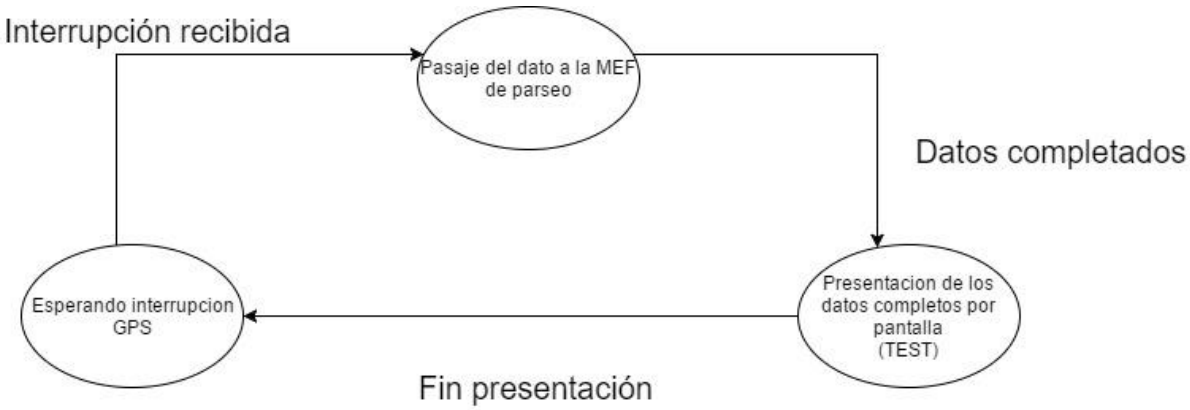


Figura 4-4

4.2.4 Pseudocódigo

```
Se recibe dato a parsear
      si el dato es válido
            caso(estado)
                  inicial:
                  si (dato == '$')
                        Inicializar variables;
                        Cambiar al estado tipo de trama ;
                  Tipo de trama:
                  si (dato == ',')
                        si ( trama == GPGGA)
                              Cambiar al estado tiempo utc ;
                        sino
                              Cambiar al estado inicial ;
                  sino
                        Obtener caracteres que indican el tipo de trama;
                  Tiempo utc:
                  si (dato == ',')
                        Cambiar al estado latitud;
                  sino
                        Obtener caracteres que indican el tiempo actual;
                  latitud:
                  si (dato == ',')
                        Cambiar al estado Norte-Sur;
                  sino
                        Obtener caracteres que indican la latitud;
                  Norte-Sur:
                  si (dato == ',')
                        Cambiar al estado longitud;
                  sino
                        Obtener carácter que indica Norte-Sur;
                  longitud:
                  si (dato == ',')
                        Cambiar al estado Este-Oeste;
                  sino
                        Obtener caracteres que indican la longitud;
                  Este-Oeste:
                  si (dato == ',')
                        Cambiar al estado inicial;
                        Activar flag de datos completados;
                  sino
                        Obtener carácter que indica Este-Oeste;
      Sino //datos invalidos
            Cambiar al estado inicial;
```

4.2.5 Planificador GPS – Test

```
Repetir siempre
      Si (flag de recepción del dato del GPS activo)
            gpsFrameParse(dato);
            limpiar flag de recepción;
      Si (flag de datos completados activo)
      Imprimir por pantalla datos obtenidos;
      Limpiar flag de datos completados;
```

4.2.6 Pseudocódigo para la arquitectura del software

La librería correspondiente al módulo GPS (realiza la comunicación serie mediante la UART 232) posee las siguientes variables:

- Enumerativos que describen el estado en la MEF
  - frameInit,
  - frameType,
  - frameUtcTime,
  - frameLatitude,
  - frameNS,
  - frameLongitude,
  - frameEO
- Estructura gpsFrameData, la cual contiene los campos de la posición global, se van completando cada vez que se recibe un carácter del gps.
  - Se define un valor para la longitud de los campo: LENGTH_FIELDS=14
  - type[LENGTH_FIELDS] : describe el tipo de trama, ellas pueden ser: GPGGA, GPGSA, GPGSV, GPGLL, GPRMC, GPVTG. En nuestro caso, tomamos la primera, GPGGA, descartando las demás.
  - utcTime[LENGTH_FIELDS]: campo en cual se va a almenar la hora en formato utc en la posición determinada.
  - latitude[LENGTH_FIELDS]: campo que almacena la latitud en la que se encuentra el GPS.
  - ns: norte/sur, en el caso que sea norte es positivo, sur negativo.
  - longitude[LENGTH_FIELDS]: campo que almacena la longitud en la que se encuentra el GPS
  - eo: Este/Oeste, este positivo, oeste negativo
- gpsCharacterReceived: variable en la cual se almacena el carácter recibido por la UART. Dicho carácter será utilizado para llenar los campos antes mencionados. Cada vez que se reciba una interrupción por la UART 232 por parte del GPS, se recibe un carácter y es almacenado en gpsCharacterReceived.
- gpsFrameIndex: los campos se deben ir completando carácter por carácter hasta la longitud de cada campo, para ello se debe tener un índice para recorrerlos. Cada vez que se cambie de campo, se debe reiniciar a cero.

- Estructura gpsFrameFlags contiene los siguientes flags:
    - characterReceived : se pone en alto en caso de que se haya recibido un carácter
    - ready: se pone en alto en caso de que se hayan completados todos los campos de la trama.

Consta de las siguientes funciones:

- void gpsFrameCharacterReceivedInit(void);
  Reinicia el valor de la variable gpsCharacterReceived en '\0'.
- void gpsFrameFlagsInit(void);
  Limpia los valores de los flags characterReceived y ready en 0.
- void gpsFrameFieldInit(char* frameField);
  Limpia el vector pasado como parámetro, colocando en cada posición del vector '\0'.
- void gpsFrameVarInit(void);
  Llama a la función gpsFrameFieldInit por cada registro de la estructura gpsFrameData, logrando limpiar todos las variables.
- void gpsSetCharacterReceived(uint8_t frameCharacter);
  Le asigna el valor pasado por parametro a la variable gpsCharacterReceived.
- uint8_t gpsReadCharacterReceived(void);
  Devuelve el valor de la variable gpsCharacterReceived.
- uint8_t gpsReadFlagCharacterReceived(void);
  Devuelve el valor del registro characterReceived perteneciente a la estructura gpsFlags.
- void gpsSetFlagCharacterReceived(void);
  Setea en 1 el registro characterReceived perteneciente a la estructura gpsFlags.
- void gpsClearFlagCharacterReceived(void);
  Setea en 0 el registro characterReceived perteneciente a la estructura gpsFlags
- uint8_t gpsReadFlagReady(void);
  Devuelve el valor del flag ready perteneciente a la estructura gpsFlags
- void gpsSetFlagReady(void);
  Setea en 1 el flag ready perteneciente a la estructura gpsFlags.
- void gpsClearFlagReady(void);
  Setea en 0 flag ready perteneciente a la estructura gpsFlags.
- uint8_t* gpsReadLatitude(void);
  Devuelve el valor de la latitude de la estrructura gpsData.
- uint8_t* gpsReadLongitude(void);
  Devuelve el valor de la longitude de la estructura gpsData.
- void gpsFrameParse(uint8_t frameCharacter);
- void gpsVarPrint(void);
  //Llamamos a la funcion uartWriteString la cual nos imprime en la hyperterminal por //comunicación serie, pasando como parámetro la uart correspondiente y la cadena o variable a imprimir:

4.2.7 Configuración de la Rutina de interrupción RS232 (UART3)

Al utilizar la librería sAPI nos facilitó la configuración de la UART3. Dicha librería nos aporta un archivo de configuración de la UART3, sAPI_Uart.c. El archivo nos brinda una función de configuración de la uart en la cual se debe especificar los puertos y el baudrate de la comunicación serie. El siguiente pseudocódigo explica la configuración:

Función de configuración de la uart (uart a configurar, baud rate):

//la uart a configurar puede ser UART232, UARTUSB o UART485
Caso UART232:

Inicializamos la uart apuntando a la uart correspondiente: LPC_USART3

Habilitamos el clock de la uart, habilitamos las FIFO y las reseteamos, deshabilitamos el Tx y las interrupciones.

Establecemos el valor del baudrate apuntando a LPC_USART3 y el valor correspondiente

Habilitamos la cola FIFO pasando la uart correspondiente

Habilitamos el transmisor Tx ya que anteriormente fue desabilitado

Establecemos el pin de la placa en la cual se genera la interrupción, en tal caso hay que establecer el SCU port, el SCU pin y la función correspondiente. Si se observa el pinout de la CIAA se puede ver que el port 2_ pin 3 (pin 87 del NXP), la función 2 corresponde a la Uart 3 Tx ; y el port 2_ pin 3(pin 88 del NXP) la función 2 corresponde a la Uart 3 Rx.

Habilitamos la interrupción apuntando a la UART 3 (LPC_USART3), habilitando el registro buffer de entrada de la interrupción.

Determinamos la prioridad de la interrupcion en 6

Habilitamos una interrupción específica del dispositivo en el controlador de interrupción NVIC apuntando a la UART3.

4.3 Conteo de Pasajeros

Para el conteo de pasajeros se emplea un esquema basada en emisores y receptores infrarrojos. Se instalan dos barreras de detección, de forma de poder discernir entre el ascenso y descenso de pasajeros, luego la cantidad de pasajeros en el micro será la diferencia entre los ascensos y descensos. El funcionamiento implementado es el siguiente:

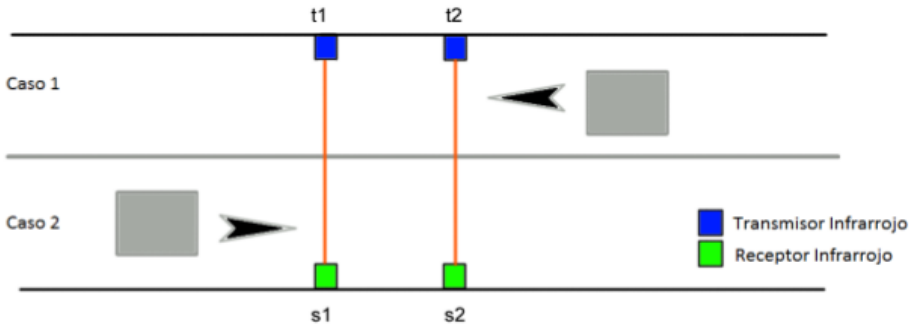


Figura 4-5

En serie con cada uno de los receptores infrarrojos, se hace una lectura con el ADC del valor de tensión. Se emplean dos canales del ADC (canal 0 y canal 1), uno asociado a cada receptor. La interpretación de los datos es la siguiente:

- Si se lee un valor alto, entonces ningún objeto está entre el emisor y el receptor
- SI se lee un valor bajo, algún objeto está situado entre el emisor y el receptor

El modelo teórico utilizado por software para el conteo de pasajeros es el siguiente:

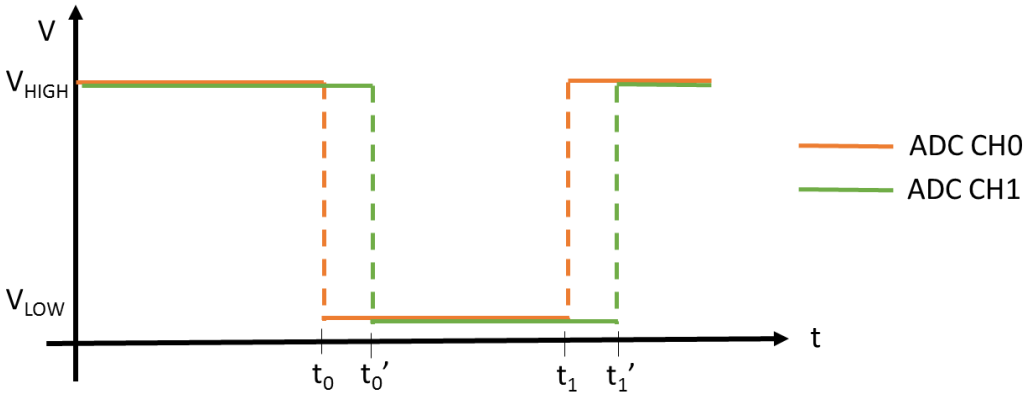


Figura 4-6

Se muestran las lecturas en cada uno de los canales del ADC (asociado a cada barrera de sensado). Cuando ningún objeto se encuentra entre el emisor y el receptor, se obtiene un nivel alto (V_{HIGH}) para ambos canales. Al atravesar una persona por los sensores (por ejemplo, para ascender al micro), se producirá la siguiente serie de eventos:

- En t_0 se interpone en la primera barrera de detección, se produce un flanco de bajada
- Conforme la persona avanza, en un instante t_0' se interpone en la segunda barrera de detección, produciendo un flanco de bajada en el respectivo canal del ADC
- En un instante t_1 deja de obstruir la primera barrera de detección (flanco de subida)
- En un instante t_1'termina de atravesar los dos sensores

El modelo teórico anteriormente planteado, para llevarse a la práctica requiere que se tengan en consideración una serie de factores:

1- El nivel alto y nivel bajo no será siempre el mismo, y las oscilaciones en las lecturas serán significativas.
2- Cada barrera de detección puede tener distintos niveles alto y bajo, de acuerdo a sutiles diferencias en la alineación
3- El modelo anterior no especifica cómo identificar personas de otros objetos (por ejemplo, si se colocan los sensores a la altura de las rodillas, la serie de eventos detallada anteriormente ocurrirá para cada pierna, generando conteos erróneos).

Por lo tanto, el modelo anterior es modificado de la forma en que se ve en la figura 4-8.

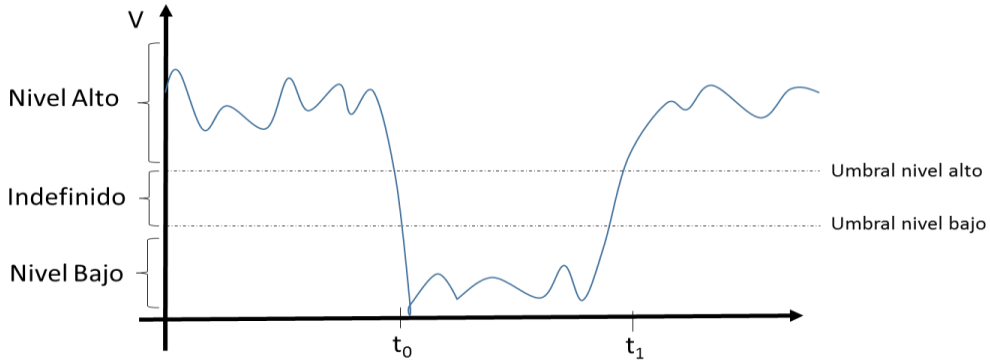


Figura 4-7

En este caso, la lectura de cada canal tendrá un "umbral nivel alto" a partir del cual se considera que no hay objetos interrumpiendo la barrera de detección, y un "umbral de nivel bajo" a partir del cual se considera que se ha producido una interrupción. Para determinar el umbral correspondiente a un nivel alto y nivel bajo, se realiza un proceso de calibración automático en cada canal. El pseudocódigo de este proceso es el siguiente:

// Se determina un promedio de lecturas para 'nivel alto' (por receptor)
Encender emisores infrarrojos
Repetir para 10 muestras
    Leer del ADC CH1 (lectura del receptor 1)
    Promediar lecturas realizadas del ADC CH0
    Leer del ADC CH2 (lectura del receptor 2)
    Promediar lecturas realizadas del ADC CH1

// Se determina un promedio de lecturas para 'nivel bajo' (por receptor)
Apagar emisores infrarrojos
Repetir para 10 muestras
    Leer del ADC CH1 (lectura del receptor 1)
    Promediar lecturas realizadas del ADC CH0
    Leer del ADC CH2 (lectura del receptor 2)
    Promediar lecturas realizadas del ADC CH1

//Calcular los umbrales para cada canal como:

$$Punto\ medio = \frac{Promedio\ nivel\ alto + Promedio\ nivel\ bajo}{2}$$

$$Umbral\ nivel\ alto = Punto\ medio + 0{,}4\ Punto\ Medio$$

$$Umbral\ nivel\ bajo = Punto\ medio - 0{,}4\ Punto\ Medio$$

Por otro lado, para identificar personas los sensores se instalan a la altura de la cintura aproximadamente, y se determina un valor de referencia para la diferencia temporal entre los flancos. Como se explicará en la sección "Ensayos y mediciones" se determina que si

$$\Delta t = |t_1 - t_1'|$$

Es menor que 100ms, no se trata de una persona (el flanco pudo haberse dado por el paso del brazo de la persona). El pseudocódigo de detección es el siguiente

Repetir

// Determina la duración de una interrupción producida en la primera línea de detección
Lectura del receptor 1 (Muestra actual CH1: Leer del ADC CH1)
Si se lee una muestra con nivel definido en el canal (nivel alto o nivel bajo)
Si la muestra actual CH1 es "nivel bajo" y la anterior "nivel alto"
Flanco de bajada CH1: Almacenar instante de ocurrencia
Si la muestra actual CH1 es "nivel alto" y la anterior "nivel bajo"
Flanco de subida CH1: Almacenar instante de ocurrencia
Si flanco de bajada CH1 > 0 y flanco de subida CH1 > 0 (se produjeron ambos flancos)
Registrar período T1
Restablecer flanco de subida y bajada CH1 a 0

// Determina la duración de una interrupción producida en la segunda línea de detección
Lectura del receptor 2 (Muestra actual CH2: Leer del ADC CH2)
Si se lee una muestra con nivel definido en el canal (nivel alto o nivel bajo)
Si la muestra actual CH2 es "nivel bajo" y la anterior "nivel alto"
Flanco de bajada CH2: Almacenar instante de ocurrencia
Si la muestra actual CH2 es "nivel alto" y la anterior "nivel bajo"
Flanco de subida CH2: Almacenar instante de ocurrencia
Si flanco de bajada CH2 > 0 y flanco de subida CH2 > 0 (se produjeron ambos flancos)
Registrar período T2
Restablecer flanco de subida y bajada CH2 a 0

// Interpreta información recolectada
Si un objeto atravesó los dos sensores (se registraron períodos T1 y T2)
Calcular diferencia ΔT entre T1 Y T2 (T1 – T2)

Si la diferencia entre flancos es mayor a 100ms
Si ΔT es mayor que cero (T1 es mayor que T2, es decir, se interrumpió primero T2)
Subió un pasajero
Sino (es menor que cero, T1 es menor que T2, es decir, se interrumpió primero T1)
Bajó un pasajero

4.4PLANIFICADOR

4.4.1 Motivación

Al empezar el desarrollo de software lo primero que se planteó fue el uso o no de sistemas operativos para la gestión de tareas y demás por parte del sistema. Dada la complejidad del proyecto, decidimos no usar sistemas operativos para así poder nosotros mismos implementar nuestro planificador de manera que sincronicemos y atendamos cada tarea según los tiempos necesarios. Esto fue gracias a que solo se cuentan con tres componentes a ser tomadas en cuenta por parte del procesador: el contador de pasajeros, el GPS y el GPRS. De esta forma se logró construir un planificador estable, seguro y eficiente.

4.4.2 Bibliotecas

Se hacen uso de las funciones correspondientes para el conteo de pasajeros, las del GPS y GPRS. Estas últimas dos modularizadas en dos archivos distintos. Se hace uso de sAPI, una librería para microcontroladores para simplificar tareas generales de uso cotidiano que funciona como una abstracción de la capa de hardware. De la librería sAPI se hace uso del tick, se configura para que existe un tick cada cinco milisegundos y enganchado a cada tick hay una función que cuenta los ticks y activa flags según el tiempo transcurrido, hay flags de cinco milisegundo y de cinco segundos.

4.4.3 Pseudocódigo

A continuación se describe el pseudocódigo para el planificador desarrollado:

Si han pasado 5 ms
    Proceso la lectura de ambos canales de los sensores
    Determino la cuenta de pasajeros
Si el GPS recibió un carácter
    Parsear la trama del GPS agregando el carácter recibido
Si el GPS recibió una trama completa lista para leer
    Activar indicador de datos listos
    Mostrar los datos del GPS
    Convertir a string la cuenta de pasajeros
Si han pasado 5 segundo y el indicador de datos listos está activo
    Si la cuenta auxiliar es menor a 3
        Incremento la cuenta auxiliar en 1
        Llamo a la MEF del GPRS pasándole los datos a enviar
    Si la cuenta auxiliar es 3, es decir, si la MEF se ejecutó tres veces
        Restablezco en cero la cuenta auxiliar
        Limpio el indicador de datos listos

Para identificar si han pasado 5 milisegundos y también 5 segundos se tiene la siguiente función en pseudocódigo invocada cada 5 milisegundos:

Activar indicador de 5 milisegundos
Si el contador interno es igual a 1000, es decir si pasaron 5 segundos
    Activar indicador de 5 segundos
Si el contador interno es mayor a 1000 y el indicador de 5 segundos está inactivo
    Restablecer el contador interno en 0

5 ENSAYOS Y MEDICIONES

5.1 CONSTRUCCIÓN DEL PROTOTIPO

Para la construcción del prototipo, una vez completadas las fases de diseño, en primer lugar, se imprime el PCB realizado con el KiCad. Para ello se utiliza papel de impresión fotográfica, y una impresora láser configurada en máxima calidad. De esta forma nos garantizamos que luego el tóner se adhiera a la placa.

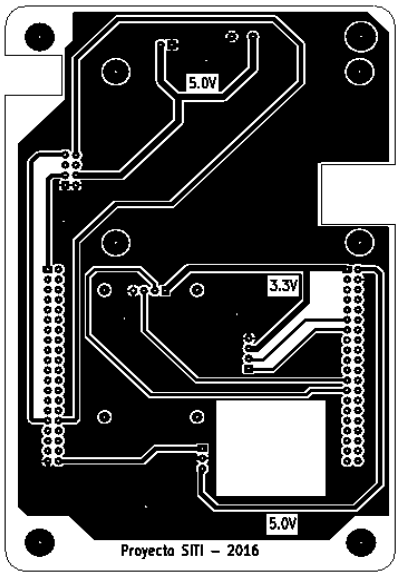


Figura 5-1

Se emplea una placa de cobre virgen de 10cm x 15cm simple faz. Para garantizar que se elimine toda la suciedad que pudiera haber sobre ella, se limpia con acetona.

Se coloca la impresión sobre la capa de cobre, y se procede a planchar de manera uniforme y constante por aproximadamente 8 minutos presionando firmemente. Luego se deja enfriar, y para remover el papel más fácilmente se sumerge en una solución de agua y jabón

Figura 5-2

Se procede a retirar el papel con la yema de los dedos para no partir el tóner que definirá las pistas. Una vez que queda una capa muy fina de papel se retira el remanente con un cepillo de dientes en desuso.

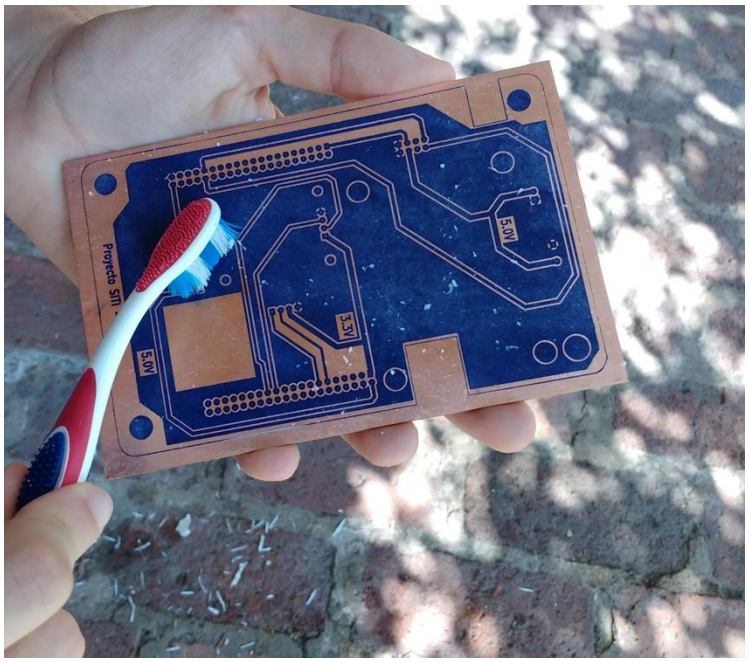


Figura 5-4

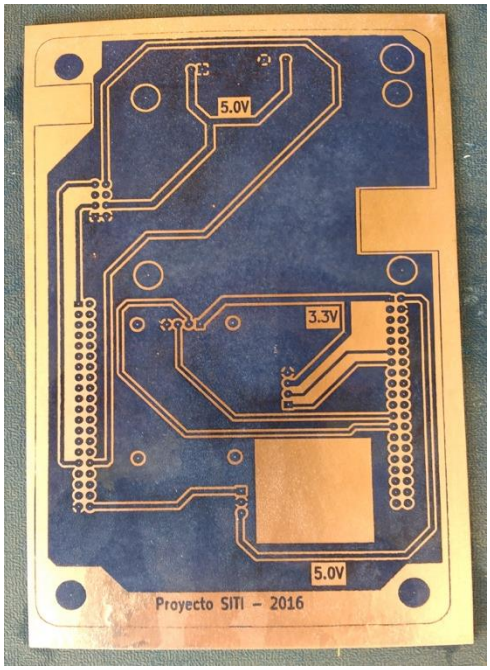


Figura 5-3

Una vez que se obtiene la placa como muestra la figura anterior, se prepara la mezcla para atacar el cobre. Se mezclan dos partes de ácido clorhídrico al 33%, con 4 partes de agua oxigenada de 100 volúmenes.

Se sumerge durante aproximadamente 5 minutos en un lugar ventilado, hasta que la solución termina de remover el cobre.

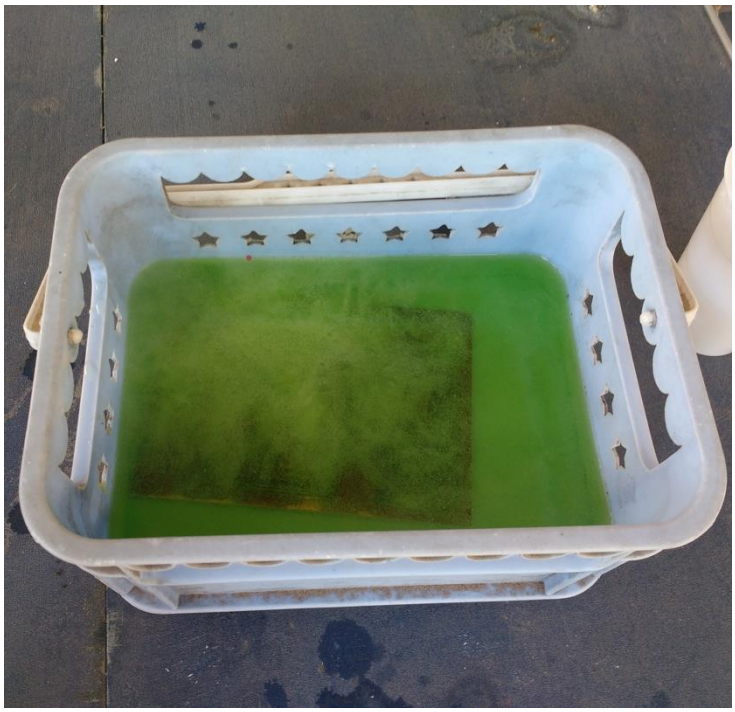


Figura 5-5

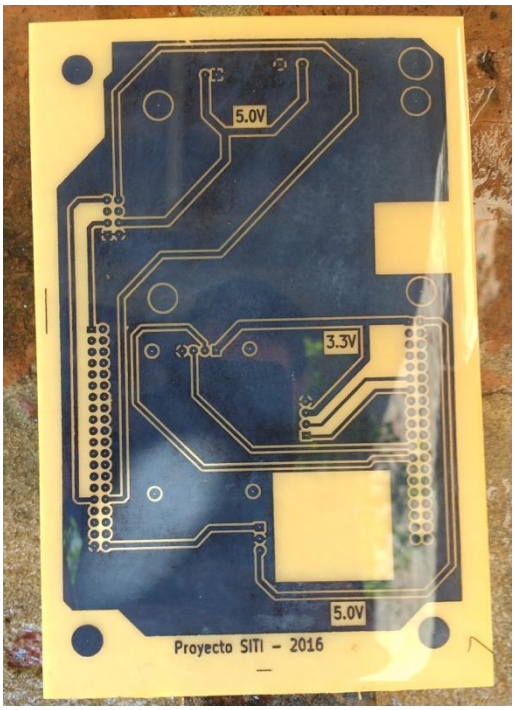


Figura 5-6

Utilizando Acetona se remueve el tóner, dejando al descubierto de esta manera las pistas de cobre de la placa.

Figura 5-7

Figura 5-8

De esta forma, ya se obtiene la placa con las pistas de cobre de acuerdo a lo definido en el diseño realizado en KiCad. Para verificar que el proceso hasta este punto haya resultado exitoso, con un multímetro se verifica la continuidad en las pistas.

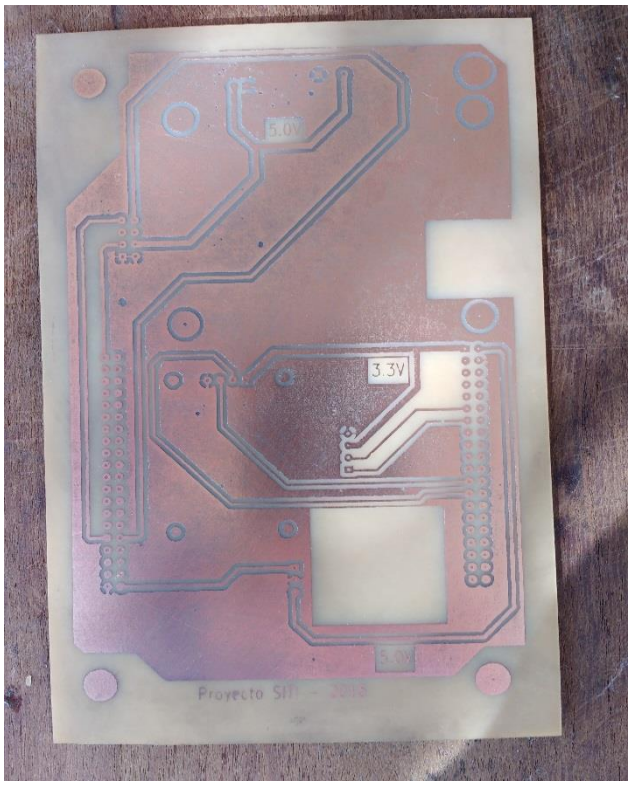


Figura 5-8

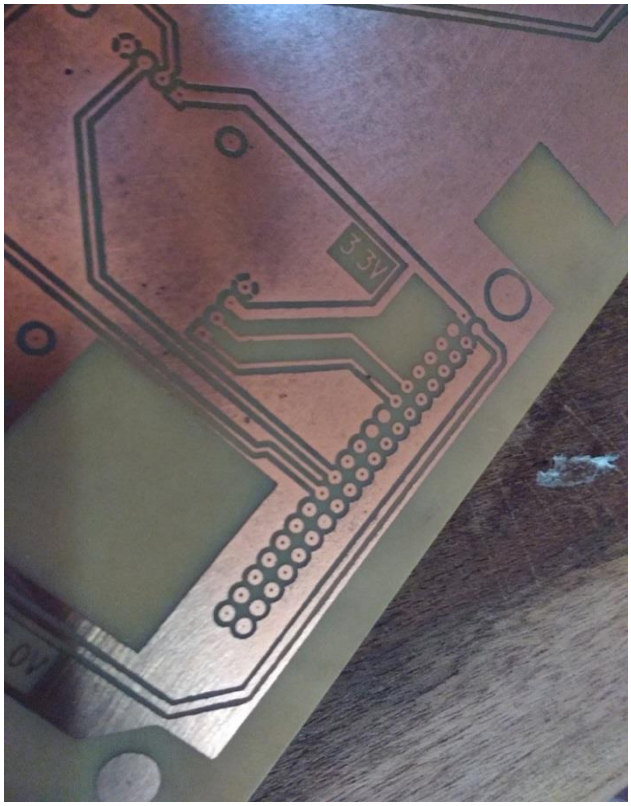


Figura 5-10

Finalmente se procede a realizar los agujeros necesarios utilizando el taladro del Laboratorio de Electrónica de la Facultad de Ingeniería, y soldar los componentes.

Figura 5-9

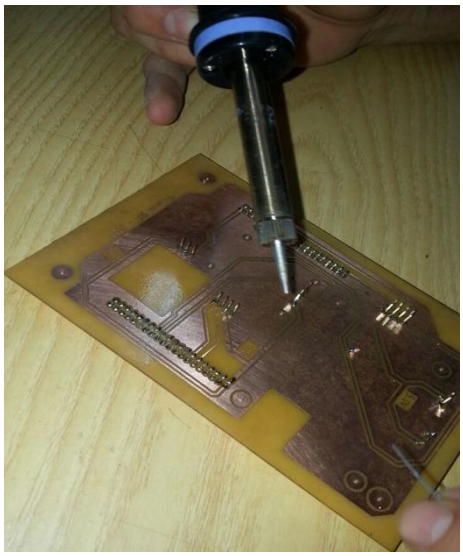


Figura 5-12

Se sueldan dos cables en la capa trasera para unificar áreas de tierra, se realizan los cortes faltantes en la palca, y como resultado de todas las etapas anteriormente descriptas se obtiene el poncho para el proyecto SITI:

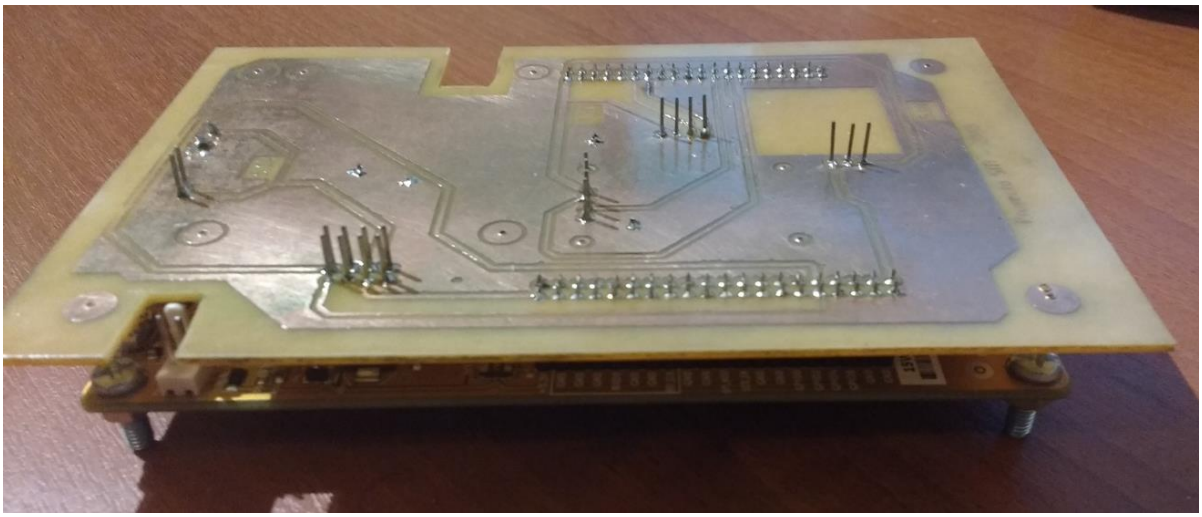


Figura 5-13

5.2 VERIFICACIÓN DE FUNCIONAMIENTO

Para verificar el funcionamiento del prototipo desarrollado, se constató el comportamiento de los diferentes módulos, trabajando en conjunto.

En lo que respecta al mecanismo de conteo de pasajeros, se utilizó un osciloscopio para medir los niveles de tensión en cada uno de los canales del ADC. Al pasar caminando, se detectaban los flancos y la salida obtenida se puede ver en la figura.

Como se puede notar, la salida responde al modelo teórico planteado en la sección "Desarrollo de Software". Se realizaron numerosas mediciones, concluyendo que un tiempo entre flancos mayor a 100ms corresponde a una persona.

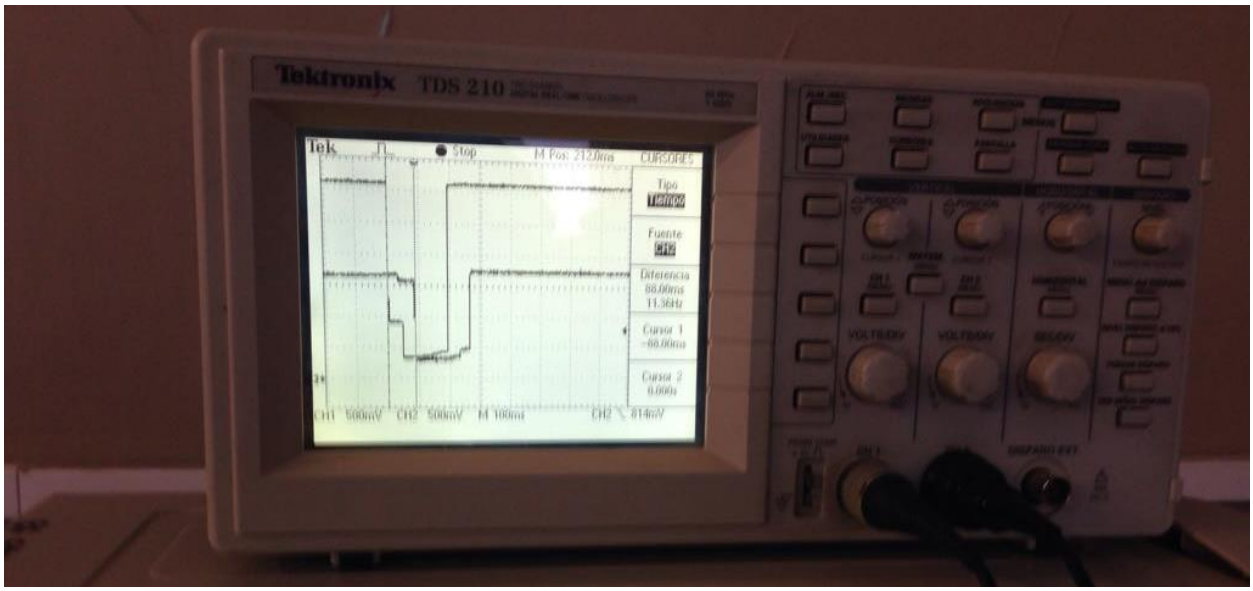


Figura 5-14

Para verificar el funcionamiento del GPS, primeramente, se utilizó la comunicación UART para ver en consola los datos de la trama recibida. Constatando con Google Maps que las coordenadas coincidan con la ubicación efectiva desde la que se realizaba la prueba, se pudo verificar el funcionamiento. Posteriormente, se empleó la comunicación GPRS para enviar los datos a la plataforma Ubidots de IoT (Internet de las Cosas), y así constatar el funcionamiento de ambos módulos.

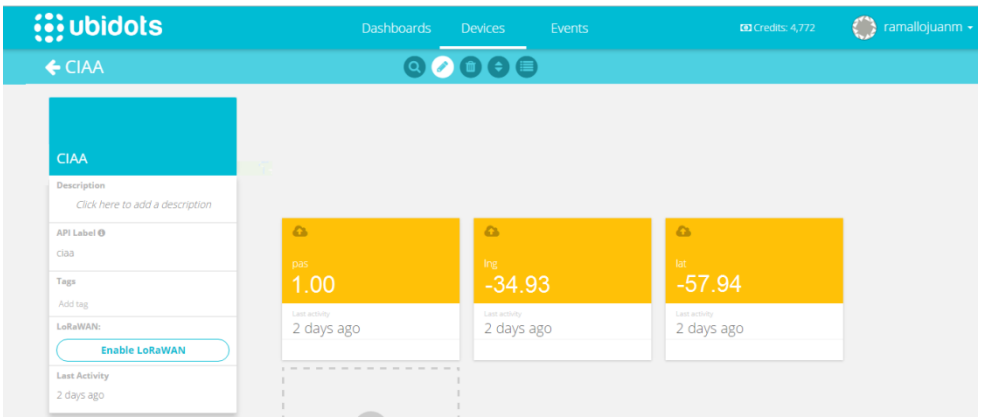


Figura 5-15

6 CONCLUSIONES

En los meses de desarrollo que se invirtieron en el funcionamiento del prototipo, pudieron completarse los objetivos fijados como primarios.

A través de sensores infrarrojos, se percibe el movimiento de cuerpos (pasajeros) en su campo de visión. Como mejora en este aspecto, se debería trabajar en un mejor montaje para garantizar la alineación emisor-receptor, ya que en ocasiones las mediciones resultan erróneas debido a problemas de sujeción.

El sistema de posicionamiento global (GPS) permite determinar la posición del vehículo de forma precisa. Cómo potencial mejora para esta funcionalidad del prototipo, se encuentra la consideración de otra antena, puesto que en ambientes interiores la señal disminuye significativamente.

En lo referido a la comunicación GPRS se logró conseguir transmisión de los datos obtenidos por los componentes anteriores. Sin embargo, la experimentación permite verificar que para numerosos puntos de la ciudad de La Plata la cobertura de telefonía móvil es sumamente deficiente, en particular en el campus de la Facultad de Ingeniería. El prototipo cumple con el objetivo planteado, pero llevado a un servicio masivo sería oportuno considerar otras alternativas de comunicación, no siendo un problema del desarrollo sino de la infraestructura de red.

Finalmente, la creación del poncho se llevó a cabo exitosamente, con algunos inconvenientes debido a la inexperiencia que pudieron ser solucionados (fundamentalmente, el innecesario uso de pistas muy finas)

El desarrollo total del prototipo insumió aproximadamente 700 horas de ingeniería en total (considerando el trabajo de los cuatro integrantes del equipo).

Concepto	Cant.	Costo Unitario ($)	Costo total ($)
Microcontrolador	1	1.200	1.200
Sensor infrarrojo	5	30	150
Módulo GPS / GPRS	1	550	550
Módulo GPRS	1	1800	1800
Horas de trabajo	700	200	140.000
Conexionado	1	200	200
		Subtotal	143.900

Tabla 6-1

La siguiente tabla especifica las tareas desarrolladas por cada uno de los integrantes. En todos los casos se asignó un responsable y un colaborador, para evitar que alguna tarea recaiga únicamente sobre una persona.

Tarea	Responsable	Colaborador
Conteo de pasajeros	Iglesias	Cobanera
Geolocalización	Pontetto	Iglesias
GPRS	Ramallo	Pontetto
Prueba conteo de pasajeros	Pontetto	Ramallo
Prueba de geolocalización	Iglesias	Ramallo
Prueba GPRS	Iglesias	Cobanera
Redacción de informes	Cobanera	Ramallo
Diseño del planificador	Pontetto	Cobanera
Diseño del poncho	Cobanera	Iglesias
Implementación del planificador	Ramallo	Cobanera
Prueba del planificador	Pontetto	Cobanera
Circuito Impreso	Cobanera	Pontetto
Soldado de componentes	Iglesias	Pontetto
Prototipo interfaz usuario	Ramallo	Iglesias
Prueba de integración	Cobanera	Ramallo
Validación	Iglesias	Pontetto

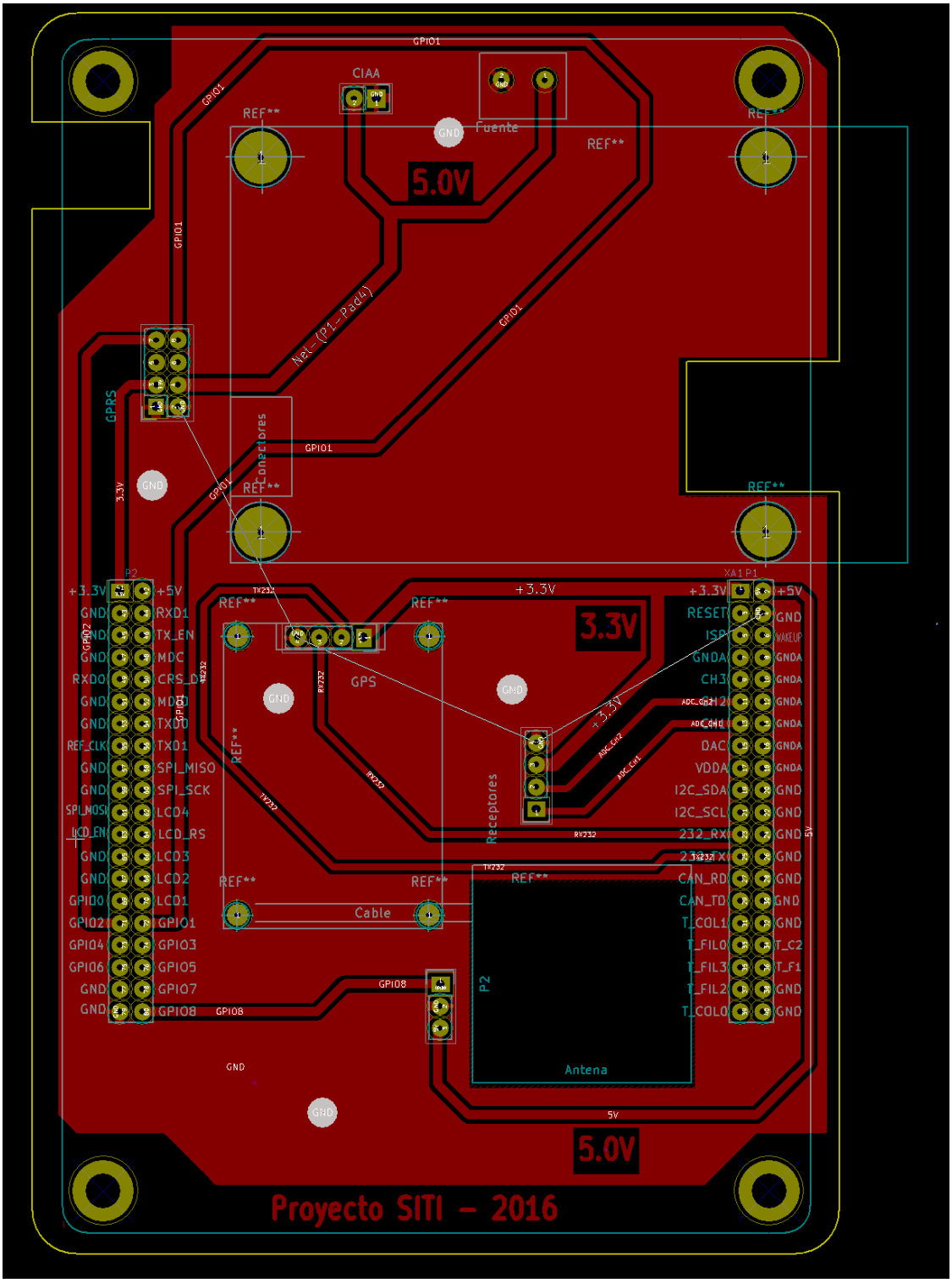
Tabla 6-2

7 BIBLIOGRAFÍA

- u-blox 6 Receiver Description Including Protocol Specification
- NEO-6 u-blox 6 GPS Modules Data Sheet
- IR333-A Data sheet. Technical Data Sheet 5mm Infrared LED
- PD333-3B/H0/L2 Data Sheet. Technical Dara Sheet 5mm Silicon PIN Photodiode
- 2n2222 Data sheet. KSP2222A NPN General-Purpose Amplifier
- Neo-6 u-blox 6 GPS Modules Data Sheet
- LEA-6/NEO-6/MAX-6 u-blox 6 GLONASS, GPS & QZSS modules Hardware Integration Manual
- LPC435x/3x/2x/1x 32-bit ARM Cortex-M4/M0 MCU; up to 1 MB flash and 136 kB SRAM; Ethernet, two High-speed USB, LCD, EMC Product data sheet
- SIM800 series AT Command Manual V1.09, 08/03/2015

8 ANEXOS

8.1 PCB



8.2 COMANDOS AT

El conjunto de comandos Hayes es un lenguaje desarrollado por la compañía Hayes Communications que prácticamente se convirtió en estándar abierto de comandos para configurar y parametrizar módems. Los caracteres «AT», que preceden a todos los comandos, significan «Atención», e hicieron que se conociera también a este conjunto de comandos como comandos AT. Midiendo la longitud de los bits se puede determinar en detalle la velocidad de transmisión.

Un aparato que implemente el conjunto de comandos Hayes se considera compatible Hayes. A partir de la versión 3.x de Windows el sistema operativo contaba con una implementación de controlador para módems compatibles con Hayes. Sin embargo, a partir de Windows 95 se desarrollaron controladores específicos para cada modem, así que la compatibilidad con Hayes dejó de ser importante y por esta razón cada vez menos módems la implementaron. Esto dificultó su uso en otros sistemas operativos, pues no resulta frecuente que haya controladores disponibles.

En la especificación de los comandos, se especifica que deberán ser enviados en mayúsculas, aunque actualmente, casi todos los proveedores de modulos GSM admiten comandos en minúsculas. El envío de comandos AT requiere la siguiente estructura:

- Petición:

- Respuesta correcta:

- Respuesta incorrecta:

La etiqueta <CR> es uno de los caracteres de control de la codificación ASCII, Unicode, o EBCDIC, que hace que se mueva el cursor a la primera posición de una línea. A veces se usa junto con el salto de línea <LF>, que lo baja a la siguiente línea; es por tanto una forma de hacer un salto de línea.