Proyecto Final: Diseño de Sistema de monitoreo de temperaturas y Posición de Personas en Lugares Públicos

Emanuel Araya Esquivel
Fabián Ramírez Arrieta
Jose David Sánchez Schnitzler
Wajib Zaglul Chinchilla

***Resumen*—En el presente documento se denota el desarrollo de un sistema capaz de monitorear las temperaturas de las personas, por medio de un programa en el lenguaje de programación ARM, sensores de temperatura, un servidor, una base de datos y una interfaz gráfica para el usuario dueño del producto. Se plantea únicamente un diseño del sistema que se desea implementar generando solamente el código de ensamblador junto a los diagramas necesarios para implementar el proyecto en la vida real.**

I. Introducción

El desarrollo de este proyecto gira en torno a buscar una solución viable al un problema específico. Este problema es buscar una manera en la que se pueda identificar en qué lugares las personas no están cumpliendo con los protocolos impartidos por el Ministerio de Salud de Costa Rica que exige que en caso de poseer síntomas del virus Covid-19 no debería salir al público, por el contrario debería permanecer en estado de cuarentena, además de realizar una prueba para saber si se posee dicho virus.

Primero se va a discutir sobre el planteamiento de la solución al problema. Una vez la idea base de la solución está bien definida, se va a empezar a seleccionar los componentes necesarios para hacer la solución de la manera más viable posible. Una vez que los componentes son seleccionados se procede a definir el tipo de conexiones que se van a hacer entre los dispositivos eléctricos y el procesador el cual va a procesar la información por medio del programa ARM.

Para la siguiente parte, se va a explicar a fondo el funcionamiento del proyecto, sus limitaciones tomando en cuenta los casos críticos en los que el prototipo pueda presentar algún fallo. Luego de que las rutas críticas están definidas se puede desarrollar más a fondo y de una manera técnica la solución planteada mostrando tipos de conexiones y los datos que se van a transportar dependiendo de cada componente.

Finalmente, se explica el proceso desarrollado en ensamblador para ARM, además se exponen las ventajas de haber escogido el procesador ARM que se especifica más adelante.

II. Planteamiento de la solución

El objetivo es centralizar la solución en un espacio público controlado. Es por esto que para minimizar errores y costos se decide implementar un sistema de lectura de temperaturas en transportes públicos como lo son los buses. El Sistema funcionará de manera peculiar ya que la idea es que se registre la temperatura de todas las personas que ingresen al bus y la ubicación en la ruta respectiva para cada persona. De esta forma se puede obtener un banco de información que pueda brindar al Ministerio de Salud de Costa Rica o del país que se implemente, un mejor sesgado de información para limitar las medidas que se deban tomar por zonas y de esta manera no perjudicar lugares que no están del todo expuestos al virus Covid-19.

Analizando a fondo el documento del proyecto se logran rescatar distintos requerimientos fundamentales para el desarrollo del prototipo que se desea implementar. De dicho documento se pueden rescatar los puntos más importantes que se indican a continuación:

- Monitoreo individual de temperatura para las personas que ingresen al transporte público.
- Monitoreo constante de la ubicación del transporte público.
- Asociar cada medición de temperatura con la ubicación del transporte público.
- Uso de un servidor para recibir y almacenar datos de esta manera dar seguimiento a los puntos anteriores.
- Debe ser un sistema autónomo, en otras palabras debe ser inteligente.
- Una interfaz gráfica para el usuario dueño del transporte público que le muestre las rutas más críticas o expuestas al virus Covid-19.

II-A. Criterios para la Elección de la Arquitectura Empleada

Para este caso en específico se optó por trabajar con arquitectura ARM M3, la razón por la cual se escogió esta arquitectura radica en el hecho que está pensada para ser utilizada en sistemas que involucren el uso de sensores [1]. Tal como es el caso para el sistema de control el cual está diseñado para que funcione por medio de sensores térmicos capaces de detectar la temperatura de todo aquel que ingrese al transporte público. Estos datos serían procesados por medio del ARM M3 y enviados a microcontrolador para luego ser enviados por medio de WiFi a un servidor que va a almacenar los datos adquiridos.

Otro motivo por el cual se implementó esta arquitectura es debido su bajo costo en consumo de potencia, lo cual es

beneficioso para un sistema que debe estar operando en todo momento. El procesador del ARM M3 ofrece un consumo de energía de 4,5 mW . Sumado a esto, se encuentra el hecho de que es una de las arquitecturas que con núcleo más reducido lo cual beneficia para que tamaño de Sistema de control sea pequeño.[2].

El procesador Cortex-M3 ha sido diseñado para ser rápido y fácil de programar, sin que sea necesario que los usuarios deban tener un conocimiento profundo de la arquitectura para crear aplicaciones simples como es el caso de este proyecto.[3].

II-B. Componentes necesarios para implementar la solución

En este apartado se van a presentar todos los componentes necesarios para implementar la solución del sistema, sus especificaciones principales y además su precio.

Para el sensor de temperatura se va a utilizar el módulo LM35. Se escoge específicamente este modelo ya que es capaz manejar de mejor manera la medición de temperaturas a largas distancias. Se puede observar en la figura 1. A continuación se muestran sus especificaciones generales.

Fig. 1: Sensor de temperatura LM35

- Voltaje de operación: 4-30 V
- Precio unitario: 2.95 dólares

Seguidamente se muestran las especificaciones del microcontrolador ARM Cortex-M3 el cual se puede apreciar en la figura 2. Es beneficioso ya que dicho microcontrolador soporta conexiones UART e I2C, entre otras como USB y SPI. Es por esto que es bastante versátil. A continuación se muestran sus especificaciones generales.

- Microcontrolador de 32 bits
- Voltaje de operación: 3.8 V
- Opera a 48 MHz
- Memoria de 256 kB
- Precio unitario: 8.53 dólares

Además, se va a necesitar un módulo GPS GY-NEO6MV2 para saber la ubicación aproximada del bus al momento de tomar las mediciones de temperatura. Dicho módulo se muestra en la figura 3. A continuación se muestran sus especificaciones generales.

- Voltaje de operación: 3-5 V

Fig. 2: Procesador ARM Cortex-M3

Fig. 3: Módulo GPS GY-NEO6MV2

- Antena cerámica
- Protocolo de conexión: UART
- Precio unitario: 2.73 dólares

Luego se va a necesitar un módulo WiFi ESP8266 ESP 201 para ser conectado con el microcontrolador ARM Cortex-M3. Dicho módulo se muestra en la figura 4. Sus especificaciones se muestran a continuación.

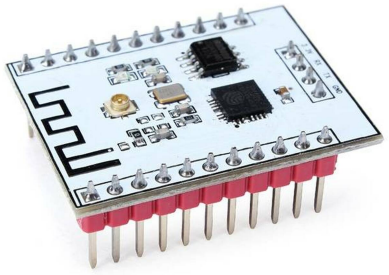

Fig. 4: Módulo WiFi ESP8266 ESP 201

- Procesador RISC integrado
- Voltaje de operación: 3.6 V
- WiFi de 2.4 GHz, con WPA y WPA2
- Protocolos 802.11 b/g/n
- Protocolo de conexión: UART

- Precio unitario: 9.32 dólares

A continuación, se muestra que va a ser necesario un microordenador como lo es el Raspberry Pi 3. Esto es necesario para adquirir la información de los módulos mencionados anteriormente para luego enviar esos datos a un servidor para ser almacenados en una base de datos. Dicho microordenador se puede apreciar en la figura 5 y sus especificaciones son la siguientes.

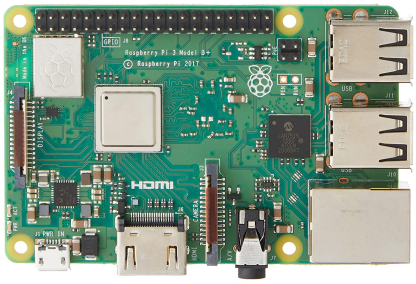

Fig. 5: Microordenador Raspberry Pi 3

- USB 2.3 y tipo C
- WiFi y Bluetooth
- Rangos de RAM: 2-8 GB
- Precio unitario: 35 dólares

II-C. Diseño de la solución al problema

Para el diseño de la solución se debe tomar en cuenta la infraestructura del transporte público. En este caso va a ser tipo bus o colectivo. Teniendo esto presente se puede afirmar que los buses presentan dos entradas o salidas, una adelante y otra atrás. Con esto en mente es fácil analizar la cantidad de materiales necesarios para implementar la solución planteada.

Primero, se va a crear un módulo que contenga el sensor de temperatura, el microcontrolador ARM Cortex-M3 y el dispositivo WiFi. Con este módulo completo se decide colocar uno tanto adelante como atrás del autobús. Con el dispositivo WiFi van a ser capaces de conectarse al microordenador Raspberry Pi 3 que se colocará únicamente uno en cada bus que se quiera instalar el prototipo.

Para finalizar se destaca que se va a hacer una conexión por medio de WiFi entre el microordenador y un servidor dedicado el cual va a poseer una base de datos para almacenar y administrar los datos recolectados. Este sería el último paso del ciclo de los datos. El sistema contemplando todos los componentes se puede apreciar en el diagrama de bloques de la figura 1.

II-D. Diagramas de Bloques

Para la realización del proyecto se diseñaron diversos diagramas e bloques para facilitar el desarrollo del mismo. De esta forma se puede visualizar la idea de diseño para ser implementada de una manera más sencilla.

En la Figura 7 se muestra el diagrama de primer nivel nivel, donde se pueden apreciar las características más generales del proyecto.

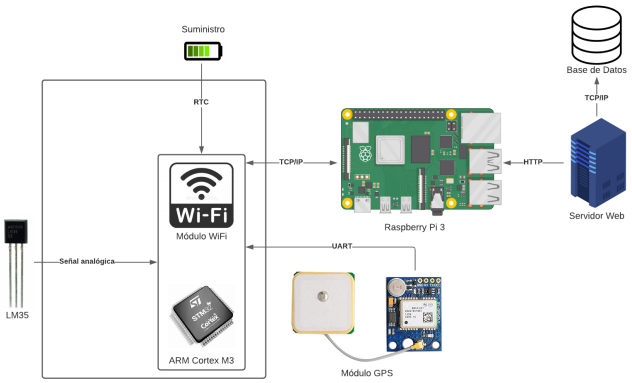

Fig. 6: Diagrama de arquitectura del Sistema de Monitoreo

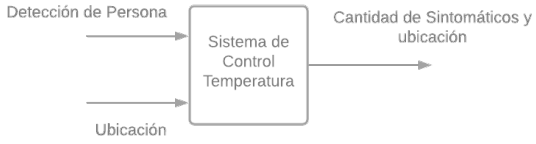

Fig. 7: Diagrama de primer nivel del diseño del sistema

En la Figura 8 se puede apreciar el diagrama de segundo nivel, en donde se muestra un poco más específico el diseño del sistema, con dos subdivisiones relevantes.

En la Figura 9 se muestra de forma mucho más específica el diseño del sistema en el diagrama de tercer nivel. En donde se pueden apreciar varios bloques de importancia con circuitos comparadores para verificar la temperatura. Es importante recalcar que se realiza la comparación de temperatura con 64 debido a que 64 representa los 38.7 °C en las unidades de entrada del microcontrolador (0 - 255). Además, se muestran otros bloques relevantes como el Raspberry Pi y el módulo GPS.

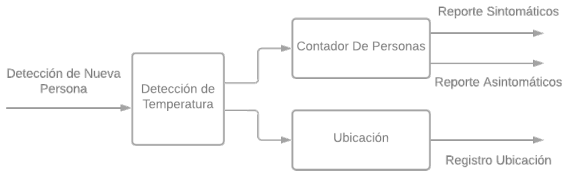

Fig. 8: Diagrama de segundo nivel del diseño del sistema

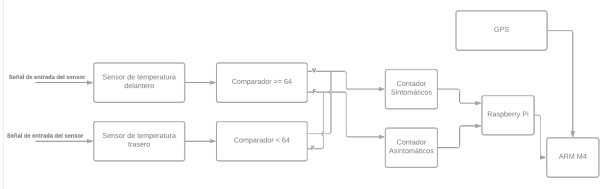

Fig. 9: Diagrama de tercer nivel del diseño del sistema

III. Funcionamiento del Sistema de Monitoreo

El Sistema de Monitoreo será colocado como un encapsulado en 2 posiciones en la parte superior del autobús al inicio del pasillo y al inicio de las escaleras traseras. De esta forma se contará con posiciones y ángulos aptos para efectuar las mediciones de temperatura. Se tendrá control sobre las personas que ingresan por la parte delantera y trasera del bus. En la figura 2 se muestran las posiciones donde se piensa colocar los sensores.

En la figura 3, se muestra el diagrama de estados, indica las diferentes tareas que deberá realizar el Sistema de Monitoreo mientras efectúa su labor de recopilar datos.

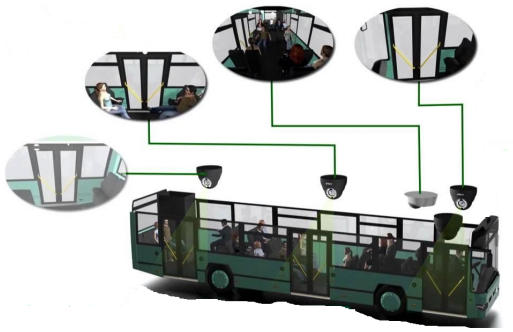

Fig. 10: Ubicación de los sensores en el autobus

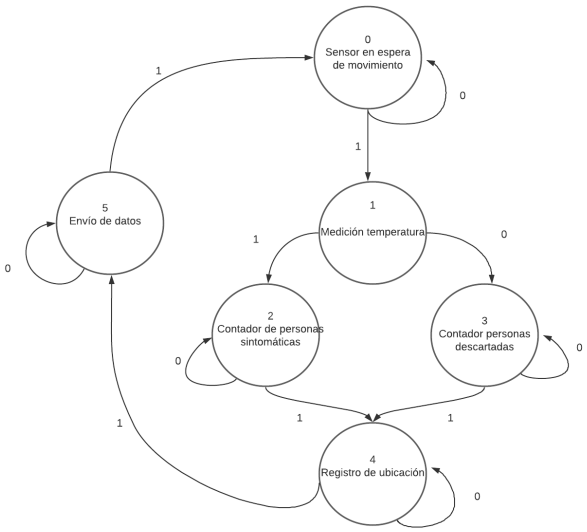

Fig. 11: Diagrama de Estados

El diagrama de flujos indica como operará el Sistema de Monitoreo. Este sistema funcionará en conjunto con el sensor de movimiento que viene incorporado en el autobús para hacer el conteo de personas que suben tanto por la puerta delantera como la trasera, de esta forma al detectarse movimiento, se enviará una señal para que el sensor de temperatura realice una medición, con el dato de la medición efectuada, este será enviado a un comparador que determina si el valor es mayor, menor o igual a 37.8 grados Centígrados. El Sistema tendrá 2 contadores para clasificar la totalidad de pasajeros entre personas sospechosas por síntomas (mediciones con valores mayores a 37.8 grados) y personas descartadas sin síntomas (mediciones con valores menores o iguales 37.8 grados). También se contará con un módulo GPS que se encargará de registrar la posición del bus al inicio de la ruta y cada vez que ocurra una parada en el recorrido del autobús hasta finalizarla. Los datos de posición y conteo de pasajeros con síntomas y sin síntomas serán enviados en tiempo real a una base de datos para ser posteriormente analizados. Una vez que los datos han sido enviados el sistema se reinicia, actualizará su ubicación y esperará a que ocurra otra parada para detectar movimiento y repetir el mismo procedimiento.

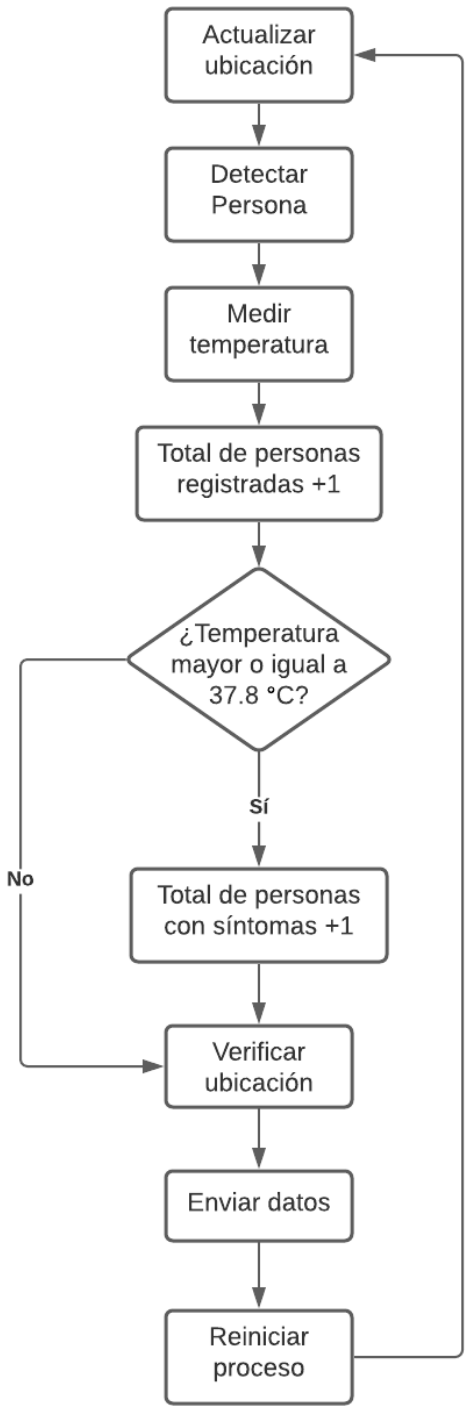

Fig. 12: Funcionamiento del Sistema de Monitoreo expresado en diagrama de flujo

Tabla I: Direcciones de memoria asignadas para el estado de banderas y valores principales.

Nombre significativo	Dirección de memoria	Valor inicial
TotalPersonas	0x20000000	0
PersonasTempAlta	0x20000004	0
DetectarPersona	0x20000008	0
UbicacionMedicion	0x2000000C	0
DiaTerminado	0x20000010	0
TempBandera	0x20000014	0
CambioTerminal	0x20000018	0
TemperaturaMedida	0x2000001C	0
MandarDatosBandera	0x20000020	0

Seguidamente se definen el uso general de los registros que se utilizaron en el desarrollo del código. Dichos registros se presentan en la tabla IV.

Tabla II: Registros a utilizar, el papel que juega en el código además de sus valores iniciales

Registro	Papel	Valor inicial
R0	Contador	0
R1	Valor de memoria	0
R2	Puntero de memoria	0
R3	Bandera para reset	0

III-A. Complicaciones en cuanto a código de ARM en Keil

En esta subsección se muestran las complicaciones presentes en el desarrollo del código de ARM en Keil. Para empezar cabe destacar que la función de Branch Link Equal la que funciona asociada a una comparación no funciona correctamente en el ambiente de trabajo de Keil por lo que hacer un salto por medio de un Link Register (LX), Branch y comparación se tuvo que resolver haciendo un bloque aparte únicamente para saltar nuevamente a donde apuntaba el registro Link Register con la instrucción BX asociado a ese registro LX. Dicha solución se puede apreciar en la figura

IV. Resultados

En este apartado se presenta la herramienta de Debugger en el ambiente de trabajo de Keil para ARM. En dicha herramienta se pueden ver los valores específicos tanto de los registros como de las direcciones de memoria. Para empezar se muestra la reserva de direcciones de memoria para los estados de las banderas y valores iniciales, esto se muestra en la figura 13.

```
3  TotalPersonas          EQU  0x20000000
4  PersonasTempAlta       EQU  0x20000004
5  DetectarPersona        EQU  0x20000008
6  UbicacionMedicion      EQU  0x2000000C
7  DiaTerminado           EQU  0x20000010
8  TempBandera            EQU  0x20000014
9  CambioTerminal         EQU  0x20000018
10 TemperaturaMedida      EQU  0x2000001C
11 MandarDatosBandera     EQU  0x20000020
```

Fig. 13: Reserva de las direcciones de memoria para estados de banderas y valores iniciales

Seguidamente se muestran los valores iniciales de los registros que se van a utilizar, esto se muestra en la figura 14. En este caso de R0 a R3 que fueron los necesarios para completar el programa, tomando como referencia la tabla IV.

```
0x00000036 F04F0000  MOV     r0,#0x00
0x0000003A F04F0100  MOV     r1,#0x00
0x0000003E F04F0200  MOV     r2,#0x00
0x00000042 F04F0300  MOV     r3,#0x00
```

Fig. 14: Inicializando los registros con sus valores de caso inicial

Luego de inicializar los registros con sus valores iniciales y además de reservar las direcciones de memoria para los

estados de las banderas y valores de los contadores se procede a crear una lógica especializada para poder implementar la solución definida en apartados anteriores.

Primero se empieza con un Reset_Handler utilizado para inicializar valores una vez que se reinicie el sistema. Esto es para el caso en el que se pase de día, se apague el bus o haya alguna falla eléctrica. Esto se puede observar en la figura 15.

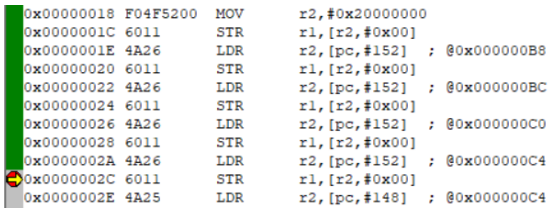

Fig. 15: Inicializando los valores para las direcciones de memoria previamente reservadas

Luego, se crea un bloque de código llamado ResetMedicion el cual es utilizado para reiniciar los estados de las banderas y valores de memoria cada vez que se termina de hacer una medición. Dicha prueba se puede observar en la figura 16.

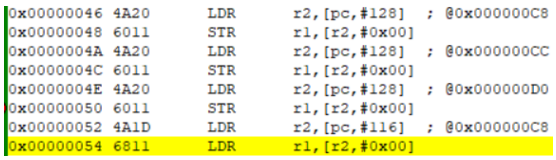

Fig. 16: Reiniciando los valores para las direcciones de memoria necesarios en cada medición

Seguidamente se actualizan valores de ubicación para verificar exactamente la ubicación en la que se está haciendo la medición y de esta manera tener un registro ordenado de temperatura ligado a una ubicación específica. Esto se logra por medio de un bloque de código llamado CambioUbicacion el cual funciona asignando un identificador dependiendo de la ubicación en la que se encuentre para después hacer la medición de temperatura donde se ligan los resultados y los datos están listos para ser enviados. Al hacer la prueba se obtuvo el resultado de la figura 17.

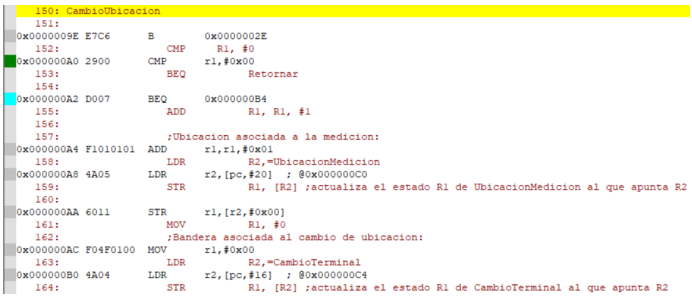

Fig. 17: Verificando la ubicación actual de la medición

Para finalizar, se hace la medición de la temperatura de la persona así que dependiendo de la temperatura que se obtenga, se va a ejecutar un bloque de código u otro. Se define la temperatura límite como 37.8 grados celsius definido por la Organización Mundial de la Salud. En caso de que la temperatura no sobrepase los 37.8 grados celsius entonces se va a sumar una unidad a la cantidad de personas registradas. En el caso de que se sobrepase la temperatura definida anteriormente aparte de sumar una unidad a personas registradas se va a sumar una unidad al total de personas sospechosas. Al final de cada bloque se levanta una bandera la cual define que los datos están listos para ser mandados y se procede a empezar nuevamente el ciclo reiniciando valores de medición hasta que se detecte una nueva persona. Dichos bloque de código se llaman TemperaturaNoAlta y TemperaturaAlta. Las pruebas que se hicieron a estos dos códigos se pueden observar en las figuras 18 y 19 respectivamente.

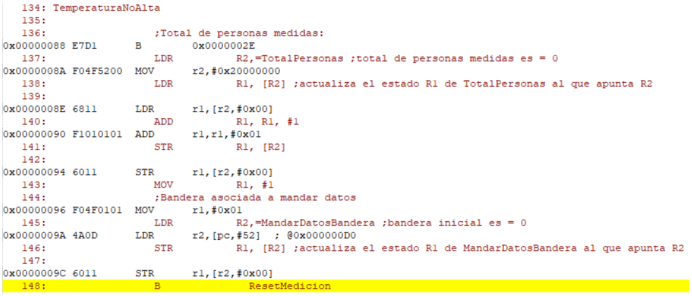

Fig. 18: Prueba para registro de personas con temperatura normal

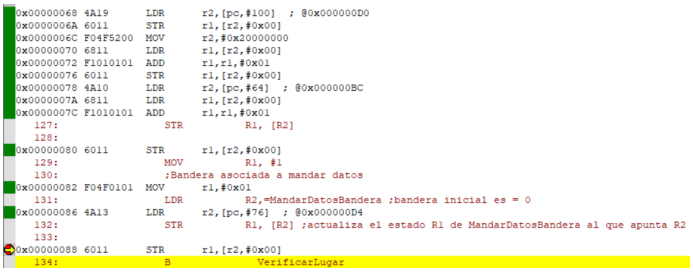

Fig. 19: Prueba para registro de personas con temperatura alta

V. Inversión económica necesaria para desarrollar el proyecto

En el apartado II-B se describió a detalle los componentes requeridos para implementar el Sistema de Monitoreo, ahora se mostrará cuanto dinero es necesario para reunir los componentes y armar el prototipo.

Tabla III: Gasto asociado a la adquisición de materia prima

Componente	Precio Unitario en $	Cantidad	Total en $
AMR Cortex-M3	8.53	1	8.53
Sensor LM35	2.95	2	5.9
Módulo WiFi	9.32	1	9.32
Raspberry Pi 3	35	1	35
Módulo GPS	2.73	1	2.73
Precio Unitario por Sistema			61.48

Considerando que para fines de este trabajo es necesario realizar 10 prototipos de este sistema, el costo de materia prima asociado tendría un valor de:

Tabla IV: Gasto asociado a la adquisición de materia prima para 10 prototipos

Componente	Precio unitario $	Cantidad	Total en $
AMR Cortex-M3	8.53	10	85.3
Sensor LM35	2.95	20	59
Módulo WiFi	9.32	10	93.2
Raspberry Pi 3	35	10	350
Módulo GPS	2.73	10	27.3
Precio 10 prototipos			614.8

Una vez establecido el costo de materia prima requerido para la construcción del Sistema, se procede a calcular los costos asociados al diseño requerido para materializar los 10 prototipos, considerando el precio que deben cobrar 4 personas con grado de bachiller en ingeniería. Según datos de Colegio Federado de Ingenieros y Arquitectos, el monto mínimo a cobrar por hora profesional para un licenciado en ingeniería es de 22753.00 colones o su equivalente a 35.47 dólares.[4]

Los cálculos de los costos asociados al diseño de los 10 prototipos serán realizados, partiendo del supuesto que el equipo encargado de realizar este proyecto pretende vender los prototipos a una empresa para la cual no trabaja de forma fija, si no que se trata de un contrato por servicios profesionales. El valor monetario será calculado en dólares puesto que es una moneda universal.

La tabla V muestra las actividades que fueron necesarias para desarrollar de forma completa el diseño de los prototipos y los tiempos que fueros invertidos estas tareas de forma colectiva.

Tabla V: Colaboradores

Actividad	Encargado	Horas
Diseño de Ingeniería	Todos	4 Horas
Diseño de Software	Todos	4 Horas
Evaluación de Prototipo	Fabián Ramírez	2 Horas
Creación de Parámetros de Prueba	Jose Sánchez	1 Hora
Reuniones y Preparación del Documento	Todos	3 Horas
Pruebas de Funcionamiento	Jose Sánchez	0.5 horas
Elaboración del Documento	Todos	10 horas

En la tabla VI se muestra la distribución de tareas en el equipo de trabajo y la cantidad de horas invertida por cada uno de los miembros para realizar cada asignación.

Tabla VI: Colaboradores

Nombre	Escuela	Horas
Emanuel Araya Esquivel	Electrónica	5 Horas
Fabián Ramírez Arrieta	Computadores	7 Horas
Jose David Sánchez Schnitzler	Computadores	7 Horas
Wajib Zaglul Chinchilla	Computadores	5.5 Horas
Total de horas laboradas		24.5 Horas

El total de horas laboradas para la elaboración del diseño es de 24.5 horas, multiplicando este valor por el precio de la hora profesional mínima tenemos que:

$$Costo_{manodeobra} = horaslaboradas * horaprofesional$$

$$Costo_{manodeobra} = \$869{,}015$$

El costo asociado a la mano de obra requerida para diseñar el prototipo del proyecto es de $869.015.
Como se mostró en la tabla IV el precio total para fabricar 10 prototipos es de $614.8

VI. Conclusiones

La elección del lugar para implementar el sistema fue una decisión fundamental en el desarrollo del proyecto, dado que un autobús es un espacio reducido y no posee muchos obstáculos es un aspecto que facilitó la identificación de limitaciones que podían afectar los prototipos. El diseño de sistemas digitales requiere de una inversión considerable en recursos humanos y económicos, situación que exige un planeamiento adecuado de los proyectos para evitar incurrir en pérdidas para las empresas. Los diagramas de funcionamiento y de flujo fueron herramientas indispensables para conseguir una idea concreta de como se deseaba que operara el sistema.Razón por la cual es una buena práctica emplear diagramas de diferentes niveles conforme se profundiza más en aspectos técnicos de un proyecto de diseño.

VII. Referencias Bibliográficas

[1] AMR, ARM Cortex M3 Datasheet.Disponible en: https://developer.arm.com/documentation/#sort=relevancy

[2] ARM, An Introduction to the ARM Cortex M3 Processor. ARM KEIL. 2006. Disponible en: https://class.ece.uw.edu/474/peckol/doc/StellarisDocumentation/IntroToCo M3.pdf

[3] Elahi. A. Sistemas Computacionales. Connecticut: Universidad del Estado de Connecticut.