



Sistema de seguretat inal·làmbric

Cristian Teniente Montilla
Tecnologies de les Telecomunicacions
Arduino

Joaquin Balletero Herguedas
Oriol Jaumandreu Sellares

Juny 2017



Aquesta obra està subjecta a una llicència de [Reconeixement 3.0 Espanya de Creative Commons](https://creativecommons.org/licenses/by/3.0/es/)

B) GNU Free Documentation License (GNU FDL)

Copyright © 2017 Cristian Teniente Montilla.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.

A copy of the license is included in the section entitled "GNU Free Documentation License".

C) Copyright

© (l'autor/a)

Reservats tots els drets. Està prohibit la reproducció total o parcial d'aquesta obra per qualsevol mitjà o procediment, compresos la impressió, la reprografia, el microfilm, el tractament informàtic o qualsevol altre sistema, així com la distribució d'exemplars mitjançant lloguer i préstec, sense l'autorització escrita de l'autor o dels límits que autoritzi la Llei de Propietat Intel·lectual.

FITXA DEL TREBALL FINAL

Títol del treball:	<i>Sistema de seguretat inal·làmbric</i>
Nom de l'autor:	<i>Cristian Teniente Montilla</i>
Nom del consultor/a:	Antoni Morell Pérez
Nom del PRA:	Joaquin Ballestero Herguedas
Data de lliurament (mm/aaaa):	<i>06/2017</i>
Titulació o programa:	<i>Grau en Tecnologies de Telecomunicació</i>
Àrea del Treball Final:	<i>Arduino</i>
Idioma del treball:	<i>Català</i>
Paraules clau	<i>Arduino, Sistema, WiFi, Seguretat</i>
Resum del Treball:	
<p>L'objectiu d'aquest projecte és el de fabricar, mitjançant la plataforma Arduino, un sistema de seguretat. Aquest producte té com a públic qualsevol llar que necessiti estar mínimament protegida. Simplement necessita una connexió inal·làmbrica(WiFi) i internet. Per a realitzar-ho s'analitzarà l'arquitectura de la plataforma Arduino, els requeriments que necessita un sistema de seguretat per a ser efectiu, i es plantejarà un disseny que cobreixi les necessitats del projecte. Amb aquest projecte es vol demostrar que un usuari amb coneixements mínims d'informàtica, pot construir el seu propi sistema de seguretat sense haver de passar per les diverses opcions que hi han al mercat, les quals són ser més costoses i restrictives. L'implementació es durà a terme mitjançant components compatibles amb la plataforma Arduino. La motivació principal del projecte ve de pensar que qualsevol llar hauria de tenir un sistema de seguretat, més enllà d'una porta amb pany, doncs actualment existeixen robatoris quan la persona no resta al seu domicili. Amb un sistema d'alarma que avisi a l'usuari, aquesta situació es pot mitigar. El projecte passarà per diverses fases, entre les quals consta d'un anàlisi de requeriments, l'intent d'elaboració d'un sistema que resulta ser inviable, una reorientació de la solució a implementar i finalment la creació i funcionament del producte construït.</p>	

Abstract:

The goal of the project is to build a security system using Arduino platform. The scope of this product is any home who needs a minium security. To perform it, only is necessary a WiFi and internet connection. to perform it, the architecture of the Arduino platform and requirements for a basic security system will be analized. After that, a design will be planned to cover project requirements. This process wants to demostrate that a low experienced user in computing can build a security system using Arduino platform, demostrating that the other options are more expensive and complex because proceed of companies. Project impementation will be performed using compatible Arduino components. The motivation proceeds of thinking about any home should have a security system, because actually sometimes are robbers come when customer is not at his home. Using an alarm system that notify user when somebody is at home, this situation can be solved. This project will proceed throught different phases, among which are requirement analysis, try to elaborate a system wich is non-viable, a reorientation of resolution to implement and finally, the creation of a working final product.

Índex

1. Introducció	8
1.1 Context i justificació del Treball	8
1.2 Objectius del Treball	9
1.3 Enfocament i mètode seguit	9
1.4 Planificació del Treball	10
1.5 Breu sumari de productes obtinguts	11
2. Elaboració del projecte	12
2.1 Anàlisi de requeriments del projecte.....	12
2.2 Obtenció del material.....	13
2.3 Esquema de muntatge	16
2.4 Iniciació de la placa Arduino	16
2.5 Implementació de la càmera OV7670	19
3. Reorientació i elaboració del projecte.....	22
3.1 Anàlisi de requeriments	22
3.2 Obtenció del sensor de moviment	24
3.3 Implantació del sensor de moviment	24
3.3.1 Muntatge de Hardware del sensor PIR.....	25
3.3.2 Programació del sensor PIR.....	28
3.3.3 Prova de funcionament del sensor PIR	28
3.4 Estratègia d'implantació WiFi	29
3.4.1 Muntatge de Hardware del mòdul Adafruit CC3000	30
3.4.2 Test de la placa Adafruit C3000	35
4. Implementació del sistema.....	37
4.1 Anàlisi de l'arquitectura del sistema a implementar.....	37
4.1.1 Anàlisi de codi del sketch de Temboo	39
4.2 Implementació de la solució de codi.....	40
4.2.1 Anàlisi del Sketch d'Adafruit CC3000	41
4.2.2 Anàlisi del codi del "Chorus" SendSMS de Temboo	43
4.2.3 Compilació del codi i execució del test (WiFi CC3000 + Tembo)	46
4.3 Implementació del sistema amb el sensor PIR	47
4.4 Compilació del codi final i test	49
4.5 Disseny encapsulament del producte.....	50
5. Proves i resultats.....	53

5.1 Demostració	53
5.2 Instruccions simplificades de muntatge del sistema	54
5.3 Resultats i viabilitat del projecte	55
6. Conclusions	59
Glossari	62
Bibliografia.....	63
Annexos	65
I.1 Iniciació a Arduino UNO i Arduino IDE	65
I.2 Inclusió de les llibreries del sistema a implementar	68
I.3 Anàlisi del codi sketch Adafuit CC3000	69
I.4 Anàlisi i deducció del codi compatible entre Temboo i el mòdul Adafuit CC3000	71
I.5 Passes per a la creació del codi SendSMS.....	76
I.6 Codi Temboo Getting Started:	78
I.7 Datasheet Arduino UNO	79
I.8 Especificacions tècniques de la càmera OV7670	80
I.9 Schematic Muntatge del sensor PIR(individual).....	82
I.100 Codi sketch de la càmera OV7670	83
Contingut adicional	86

Llista de figures

Figura 1. Diagrama de Gantt principal del projecte	11
Figura 2. Component Adafruit WiFi CC3000 a) primera cara. b) segona cara..	14
Figura 3. Placa Arduino UNO	15
Figura 6. Càmera OV7670	15
Figura 14. Esquema de muntatge test de la placa Arduino.....	17
Figura 15. Demostració de funcionament de concepte Arduino UNO.....	18
Figura 18. Esquema de muntatge càmera OV7670 amb Arduino UNO [8].....	19
Figura 19. (1) Fotografia de muntatge de la càmera OV7670 amb Arduino UNO	19
Figura 20. (2) Fototrafia de muntatge de la càmera OV7670 amb Arduino UNO .	20
Figura 21. Imatge d'internet d'un sensor PIR HC-SR501.....	24
Figura 22. Fotografia d'un sensor PIR HC-SR501. Part frontal.....	25
Figura 23. Fotografia d'un sensor PIR HC-SR501. Part darrera	25
Figura 24. Esquema de muntatge de la prova de concepte amb el sensor PIR ...	26
Figura 25. Fotografia del muntatge físic de la prova de concepte del sensor PIR	27
Figura 26. Demostració de la prova de concepte del sensor PIR quan no detecta moviment	28
Figura 27. Demostració de la prova de concepte del sensor PIR quan detecta que hi ha moviment	29
Figura 28. Fotografia del component WiFi Adafruit CC3000	30
Figura 29. a) Fotografia del component WiFi Adafruit CC3000 soldat a un array de connectors i introduït a una protoboard CC3000. b) Fotografia ampliada del component Adafruit CC3000 connectat a una protoboard.	30
Figura 30. Imatge d'un esquema de muntatge del component Adafruit CC3000 amb Arduino UNO utilitzant una protoboard [12].....	31
Figura 31. Esquema de muntatge realitzat utilitzant el sensor PIR, Arduino UNO i el mòdul Wifi Adafruit CC3000 per a crear el sistema final del projecte	33
Figura 32. Imatge del sistema final construït (Hardware).....	34
Figura 33. Sketch que mostra el codi implementat de la connectivitat amb una placa Arduino UNO i el component Adafruit CC3000	35
Figura 34. Captura de la sortida Serial d'Arduino UNO amb el Sketch "built" utilitzant el component Adafruit CC3000	36
Figura 35. Portal principal d'usuari de la pàgina web de Twilio [15]	38
Figura 36. Esquema conceptual de l'arquitectura del sistema a implementar ..	38
Figura 37. Demostració de funcionament del codi del sistema implementat utilitzant el sketch SendSms adaptat al nostre hardware.....	46
Figura 38. Demostració de recepció d'un SMS utilitzant el sketch generat.....	46
Figura 39. Demostració final del sistema implementat per al projecte.	49
Figura 40. a) Fotografia d'un mòdem Vodafone antic en desús. b) fotografia d'un mòdem Vodafone antic obert i buit.	51

Figura 41. a) Mòdem Vodafone amb els components del sistema del projecte introduïts. b) Mòdem Vodafone amb els components del sistema del projecte introduïts mostrant el sensor PIR.	51
Figura 42. Sistema final implementat i encapsat sense el sensor PIR cobert...	52
Figura 43. Producte final. Demostració completa. a) fotografia 1. b) fotografia 2.	53
Figura 8. Pàgina web de descarrega d'Arduino IDE [6]	65
Figura 10. Selecció de placa en el programa Arduino IDE.....	66
Figura 11. Selecció de port USB de sortida de Arduino IDE	67
Figura 12. Demostració de compilació de codi en Arduino IDE	67
Figura 40. Captura de pantalla de les llibreries utilitzades per el component Adafruit CC3000	68
Figura 45. Exemple de generació de codi del Sketch SendSMS utilitzant el portal Temboo[14].....	77
Figura 4. Datasheet Arduino UNO [4].....	79
Figura 5. Esquema de connexions Arduino UNO [4].....	79
Figura 7. Datasheet del component OV7670	80
Figura 16. Definició dels pins del component OV7670 [7].....	81
Figura 38. Datasheet realitzat utilitzant el sensor PIR, Arduino UNO i el mòdul Wifi Adafruit CC3000 per a crear el sistema final del projecte	82

Taula de contingut del codi utilitzat en el projecte

Codi 1. Primer test Arduino UNO.....	17
Codi 7. Prova de concepte del sensor PIR.....	28
Codi 2. Exemple d'iniciació de Temboo.....	39
Codi 3. Sketch BuiltTest Adafruit CC3000	41
Codi 4. Sketch del sistema de seguretat del projecte sense l'utilització del sensor de moviment.	45
Codi 5. Sketch final del projecte. Sistema de seguretat.	48
Codi 2. Exemple d'iniciació de Temboo.....	78

Taules i gràfics

Taula 1. Proves del producte	55
Gràfic 1. Probabilitat de fallada del sistema.....	56
Gràfic 2. Senyal de la placa segons distància.....	57
Taula 2. Compliment de requeriments inicials	58

1. Introducció

1.1 Context i justificació del Treball

Actualment, es creu que per a poder monitoritzar allò que hi ha a la llar de forma remota, un ha d'adquirir una sistema de videovigilància inal·làmbric, comprant-lo a una botiga física o per internet. Es parteix amb l'objectiu de demostrar que la hipòtesi anterior es incorrecta. Es vol obtenir un sistema de videovigilància inal·làmbric, sense haver de comprar un amb les limitacions de software i costos que comporta adquirir-lo. Es posa en manifest l'importància de que qualsevol llar domèstica tingui instal·lada un sistema de vigilància per a la seva seguretat, sent capaç controlar en tot moment la situació al domicili.

Si analitzem el mètode d'obtenció del producte final, es poden observar preus similars en quant a construir-lo o fabricar-lo, veient com per uns 40€ aproximadament pots adquirir càmeres IP de videovigilància, per exemple. La diferència s'aprecia en la resta de variables, es a dir, les càmeres comercials venen amb el seu software preinstal·lat i per a fer-la servir, o es devia l'obtenció de vídeo cap a un servidor extern mitjançant internet o es crea una xarxa privada virtual (VPN), amb el treball i coneixements previs que això comporta . Amb Arduino, les possibilitats de configuració les crea l'usuari, podent adaptar les funcions del dispositiu segons les seves necessitats.

La segona observació és que la majoria de càmeres IP estan pensades per a visualitzar la llar domèstica des de la mateixa xarxa, llavors si es vol crear una connexió remota fora del domicili, les condicions econòmiques canvien substancialment: hi han empreses que es dediquen a configurar la teva xarxa per a crear una VPN que sol ser monitoritzada per ells i així obtenir la connexió desitjada, i és quan l'usuari, per a obtenir aquest servei, ha de pagar una quota mensual o anual, a part de comprar la càmera. Aquest projecte pretén utilitzar el codi open source i hardware d'Arduino per a crear i configurar un sistema de seguretat connectat amb l'usuari mitjançant connexió per internet.

1.2 Objectius del Treball

- Eliminar la hipòtesi de que per a poder monitoritzar el que passa a un domicili, un hagi de comprar un producte comercialitzat, demostrant que hi han altres opcions amb més possibilitats.
- Demostrar que actualment, hi han molts models físics de sistemes de seguretat, però cap d'aquests es proveeix de la configuració necessària per a crear una xarxa privada amb un cost econòmic, deixant a questa tasca per a empreses que distribueixen i configuren aquest servei, oferint una quota mensual o anual. Aquest projecte pretén obtenir aquest servei de manera privada i sense cap tipus de cost.
- Posar en manifest la facilitat de fabricació d'un sistema de seguretat domèstic, demostrant que al ser 'open source' es pot configurar a mida de les necessitats de l'usuari.
- Evidenciar que actualment fabricar i configurar un dispositiu electrònic no requereix de tanta logística i coneixement com fa alguns anys, doncs diverses plataformes com Arduino, Raspberry, etc. han anat sortint al mercat oferint àmplies possibilitats inexistents fins fa pocs anys.

1.3 Enfocament i mètode seguit

Es pretén realitzar les següents fases que resultaran en l'objectiu principal del projecte:

- Primerament, l'anàlisi per a decidir quin sistema de seguretat es vol crear per a cobrir les necessitats del projecte.
- Segon, l'obtenció i fabricació del hardware necessari per a poder construir el producte desitjat. Es pretén que l'adquisició dels components sigui el menys costós possible per a guanyar accessibilitat de cara a l'usuari final, es a dir, el consumidor del producte.
- Un cop acabat el primer apartat, es realitzarà la programació de la placa Arduino per a que faci la funció desitjada, estudiant quin codi s'adaptarà millor a la placa per al l'objectiu d'aquest projecte. Aquest apartat serà el més extens ja que ocupa tot el procés de fabricació del producte, proporcionant-li funcionalitat
- Finalment, i un cop resolt els tres primers apartats, es procedirà a la creació i implantació d'un mètode de connexió, amb el que l'usuari es podrà connectar al dispositiu des de fora de la seva xarxa.
- Amb el producte implementat, es procedirà a demostrar la hipòtesi del projecte.

1.4 Planificació del Treball

En els pròxims capítols s'exposarà el procés d'obtenció d'una solució que satisfagui l'objectiu del projecte, detallant pas a pas la metodologia realitzada a cada part del recorregut.

- Es començarà per l'anàlisi del sistema a implementar i el seu hardware requerit, l'obtenció d'aquest, possibles alternatives i finalment es plantejarà un esquema de muntatge. (2 setmanes)
- Seguidament, es procedirà al la programació de la placa per a que realitzi la funció adequada, passant per la valoració de quin llenguatge s'utilitzarà i quin codi s'ha d'afegir per a implementar totes les accions necessàries. (2-3 setmanes).
- Un cop s'obtinguin els dos passos anteriors, es procedirà a la part més extensa del projecte: l'implementació del codi i hardware. També s'inclouran les proves de test realitzades i els possibles contratemps o canvis d'estratègia per a cobrir les necessitats del projecte. Tota la metodologia serà detallada.(4-5 setmanes)
- Finalment es procedirà a la creació de la xarxa per a que el sistema de seguretat tingui connectivitat amb l'usuari des de qualsevol lloc amb internet.
- L'objectiu final quedarà cobert si s'aconsegueix establir comunicació entre el dispositiu fabricat dins de la llar i un dispositiu mòbil fora de la mateixa xarxa. (1 setmana)
- Amb tot acabat ja es tindrà tota la informació necessària per a poder procedir amb la hipòtesi i es realitzarà una comparativa punt per punt entre el producte creat i un de comercial. (1 setmana)

La duració total del projecte, tenint en compte imprevistos, no hauria de sobrepasar les 13 setmanes, sent 12 setmanes el temps ideal, es a dir, 3 mesos aproximadament.

S'ha generat un Diagrama de Gantt il·lustrant la planificació temporal del projecte seguint les pautes de l'apartat anterior (fig. 1).

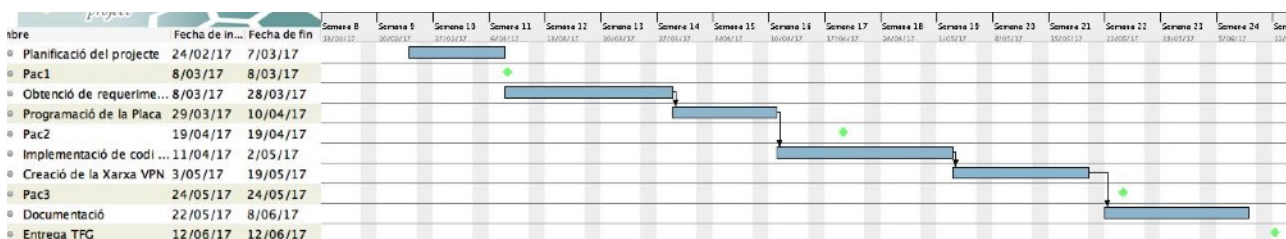


Figura 1. Diagrama de Gantt principal del projecte

Per a elaborar els esquemes de muntatge es farà servir una aplicació anomenada Fritzing [16], que ens ajudarà a planificar el muntatge de hardware de tots els components.

1.5 Breu sumari de productes obtinguts

- Placa Arduino UNO rev.3.
- Placa WI-FI Adafruit CC3000 WiFi Breakout with Onboard Ceramic Antenna
- Adaptador Aukru® 9V 1000mA alimentador Cargador 5.5mm x 2.1mm 1.6
- Cable USB tipus A a tipus B
- Mòdul càmera OV7670 640 x 480 VGA CMOS
- Kit de components de cables, condensadors, resistències, LEDs, etc.

2. Elaboració del projecte

2.1 Anàlisi de requeriments del projecte

Existeixen diverses solucions que cobreixen l'objectiu del projecte, que és el proporcionar a l'usuari d'un sistema de seguretat el qual informi si una persona no autoritzada entra a la seva llar.

Per a això es necessita que el sistema sigui:

- **Immediat:** en quant es detecti a un usuari no autoritzat aquest haurà d'avisar al receptor immediatament, doncs un sistema de seguretat que trigui en avisar no es útil. El temps d'avís no ha de sobrepassar els **2 minuts**.
- **Fixe:** el sistema s'haurà de situar en un lloc concret de la llar on es pretengui monitoritzar l'activitat, com pot ser a l'entrada de casa, a la terrassa, etc. Ha de poder estar alimentat permanentment a la corrent sense que s'apagui. **El percentatge de probabilitat de fallada degut a problemes en l'alimentació del sistema no haurà de major al 2%.**
- **Connectat permanentment:** per a que el sistema funcioni durant un període de temps acceptable per a que sigui útil, s'haurà de poder connectar a gust de l'usuari de manera indefinida. Per a mesurar l'acompliment d'aquest requeriment, es voldrà que el sistema estigui connectat sense interrupcions **com a mínim durant 15 hores seguides**. S'ha de tenir en compte que la xarxa no pot tenir problemes de connectivitat, ja que la placa Arduino no es podrà re-connectar si no es apagant-la i encenent-la manualment.
- **Fiable:** el sistema haurà de ser suficientment senzill com per a reduir al mínim els errors de hardware i de software que facin que el sistema tingui fallades de connectivitat. Per a mesurar aquest factor, es defineixen dos variables. Per un costat la probabilitat de fallada del sistema, i per altre, el numero de fallades per dia que pot tenir. Per a aconseguir aquest requeriment es voldrà, tenint en compte la construcció i la plataforma, en primera instància, que el sistema tingui un **percentatge de fallada d'un 10% quan s'utilitzi**.
- **Fàcil de fer servir i construir:** l'usuari final no te per que ser expert amb el sistema, amb el que el seu ús ha de ser el més senzill possible, d'aquesta manera arribarà a un public més ampli. Es preveu que **la duració del muntatge no hauria de sobrepassar les 3 hores**, sinó no es còmode per l'usuari.
- **Fàcil d'instal·lar:** el sistema ha de ser suficientment fàcil d'instal·lar com per a solament haver-lo de fixar a una paret i connectar-lo a la corrent. **La instal·lació no ha de suposar cap mena de despesa econòmica**, podent reutilitzar material reciclat per a encapsar i col·locar el sistema.
- **Inalàmbric:** el sistema no ha de requerir de gaire connectivitat amb cable, solament el de corrent. La transmissió d'informació cap al router de casa es farà per WiFi amb una **freqüència de 2.4GHz**, que es la compatible amb la

gran majoria de routers al mercat, ja que es un estàndard definit. **La distància màxima a la que ha de poder estar col·locat respecte un punt d'accés WiFi és d'uns 5 metres.**

Un cop finalitzat l'implementació del producte, es comprovarà si reuneix tots els requeriments descrits.

Amb aquests requeriments es plantegen dos solucions:

1. Una opció es crear, mitjançant un mòdul de càmera i un mòdul WiFi compatibles ambdós amb Arduino, crear una càmera de videovigilància IP, la qual ens envii una imatge captada per la càmera del moment en el qual li fem una petició.
2. Un altre sistema a implementar seria, mitjançant un sensor de moviment i un mòdul WiFi compatible amb arduino, la creació d'un sistema que capti el moviment d'un usuari i que t'avisí quan ha captat presència via web.

Aquests dos possibles sistemes proposats han de complir amb les especificacions dels requeriments del projecte i s'han de poder implementar respectant el temps del qual es disposa en aquest projecte.

Finalment s'escull com a seleccionat la creació d'una **càmera de videovigilància IP.**

Resum de funcionament de la solució escollida:

Aquesta càmera haurà de capturar una imatge en el moment que detecti que hi ha alguna presència, i enviar-se-la a l'usuari mitjançant un correu electrònic o un mecanisme similar per a que sàpiga que ha passat en aquell moment. L'anàlisi de l'enviament de la imatge es realitzarà a posteriori, doncs primerament s'intentarà construir una càmera que capturi imatges, i que les pugui enviar inal·làmbriament per internet. A més, s'ha de poder controlar des de fora utilitzant una VPN (virtual private network), des de la qual l'usuari podria veure una imatge en el moment de fer la petició.

Els pròxims apartats detallaran la creació del sistema partint de les especificacions del projecte.

2.2 Obtenció del material

Per a obtenir el hardware requerit, primerament s'han buscat referències de projectes similars existents per a verificar que la compra realitzada dels components es l'adequada.

Realitzant recerca per internet es troben diversos projectes existents amb funcionalitats finals buscades en aquest projecte:

- Projecte d'una càmera connectada per ethernet [1]
- Projecte de fabricació d'una càmera de seguretat amb arduino YUN [2]

Posem especial atenció al següent projecte: Wireless Camera with Arduino and the CC3000 WiFi chip [3]

Per a implementar la connexió del sistema amb el router hi han dos opcions:

- Utilitzar cablejat ethernet entre el sistema i el punt d'accés a internet. El cable ofereix velocitat de connexió de fins a 300Mbps depenent del servei contractat, però el sistema construir no necessita gran transmissió de dades, ja que simplement es vol capturar una imatge, que serà d'uns 100Kbs. A més es sacrifica un dels requisits, que és que el sistema sigui sense fils.
- Utilitzar una placa WiFi integrada en el sistema per a realitzar la connectivitat amb el punt d'accés. Amb aquesta solució es compleixen els requeriment, i avui dia una connexió WiFi estàndard pot oferir velocitats mínimes de 20Mbps, més que suficient per a transferir informació entre l'usuari i el sistema.

Amb aquest escenari, s'ha decidit adquirir un modul Wi-Fi en comptes de l'Ethernet, i investigant, l'única placa compatible amb el mòdul Wi-Fi és l'escollida al projecte, l'Arduino UNO.

El mòdul Wi-Fi es: Adafruit CC3000 Breakout Board. Aquest component es l'únic trobat amb connectivitat inal·làmbrica dins de les funcions requerides del projecte.

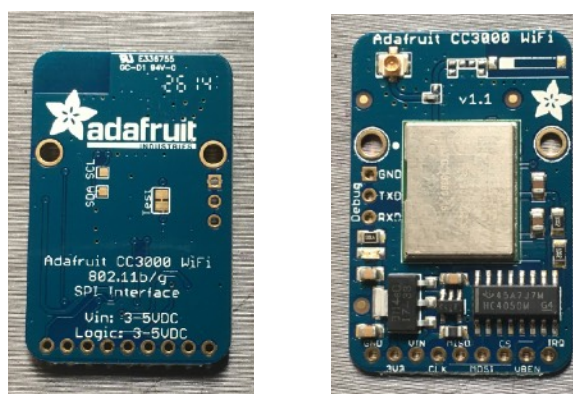


Figura 2. Component Adafruit WiFi CC3000 a) primera cara. b) segona cara

El cost de la placa WiFi ha sigut d'uns 30€
La placa principal escollida per a treballar amb la resta de components es el model Arduino UNO rev 3 (fig. 3).



Figura 3. Placa Arduino UNO

El cost de la placa Arduino UNO ha sigut de 24€.

Finalment, en quant a components principals, falta seleccionar quin modul de càmera utilitzarem, i el seleccionat es aquest, que a part de complir la seva funció és molt econòmic: OV7670 640 x 480 VGA CMOS Camera:

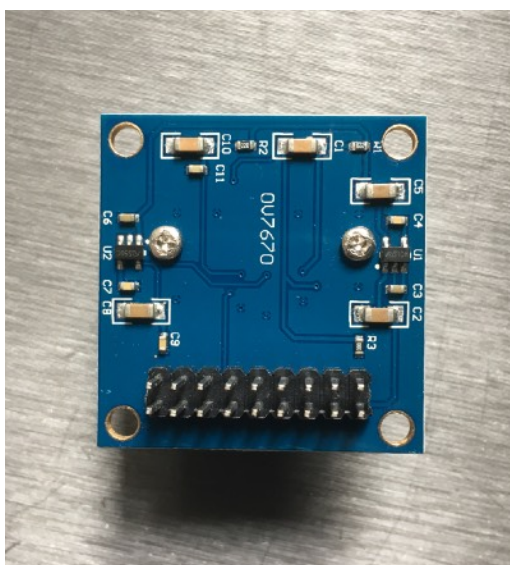
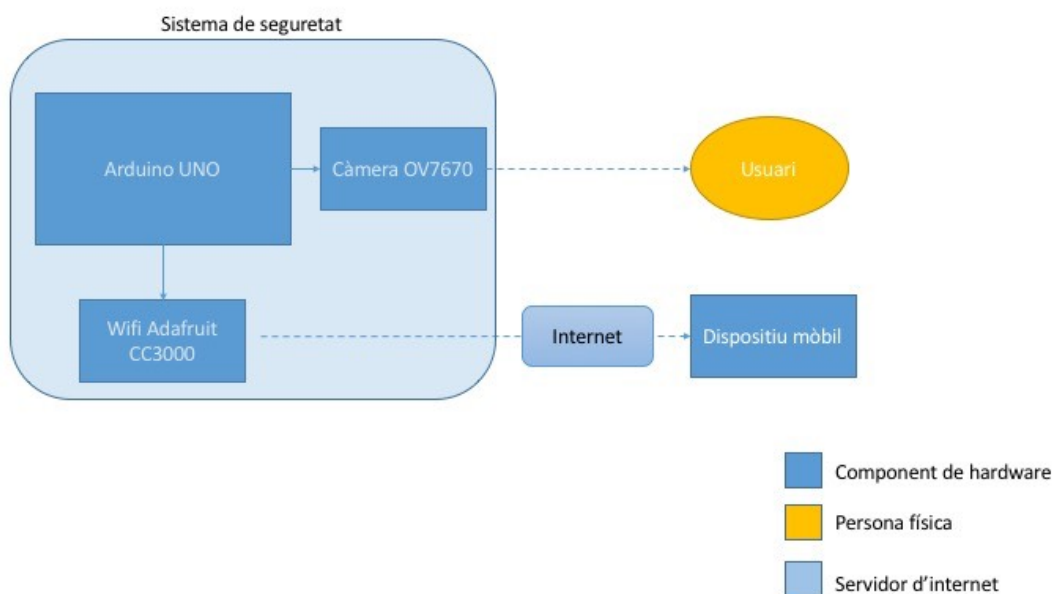


Figura 6. Càmera OV7670

El cost de la càmera ha sigut de 7€.

2.3 Esquema de muntatge

Un cop decidit els components, es procedeix a mostrar un possible esquema funcional amb diversos components connectats.



2.4 Iniciació de la placa Arduino

Es realitza un test de funcionament de la placa Arduino UNO per a familiaritzar-se amb el seu entorn.

La pàgina oficial d'Arduino[6] ens mostra com inicialitzar la placa un cop obtinguts els components bàsics.

A continuació es realitza un muntatge simplificat per a provar la compilació de codi a la placa que constarà d'un led connectat a la placa que s'il·luminarà i s'apagarà cada segon. [Annex I.1]

L'esquema de muntatge mostra com aniran connectats els diversos components que participen en aquest test(fig.14).

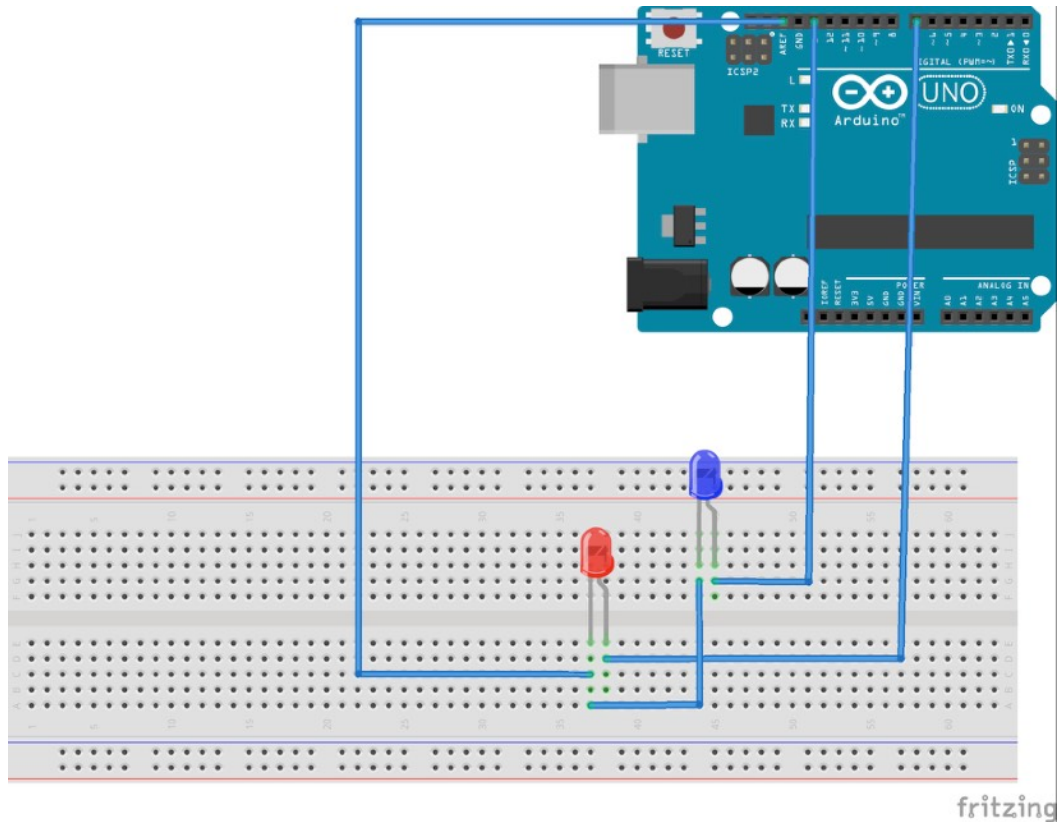


Figura 14. Esquema de muntatge test de la placa Arduino

Per a això s'ha de programar la placa amb el següent codi per a que realitzi la seva funció:

```
//creem una variable per a la sortida numero 13 que li anomenem "led"
//tindrà valors de nombres enters:
int led=13;
int led2=7;
void setup() {
  // put your setup code here, to run once:
  //li diem que el led numero 13 tindrà funció de sortida
  pinMode(led,OUTPUT);
  pinMode(led2,OUTPUT);
}
void loop() {
  // put your main code here, to run repeatedly:
  // la sortida tindrà voltatge màxim (la llum s'encendrà):
  digitalWrite(led, HIGH);
  digitalWrite(green, LOW);
  //esperem 1 segon
  delay(500);
  digitalWrite(green, HIGH);

  //la sortida tindrà voltatge mínim (la llum s'apagarà)
  delay(500);
  digitalWrite(led, LOW);
  digitalWrite(led2, LOW);
  //tornem a esperar 1 segon:
  delay(500);
  digitalWrite(led2, HIGH);
  delay(500);
}
}
```

Codi 1. Primer test Arduino UNO

Es verifica el funcionament de la placa Arduino UNO (fig. 15).

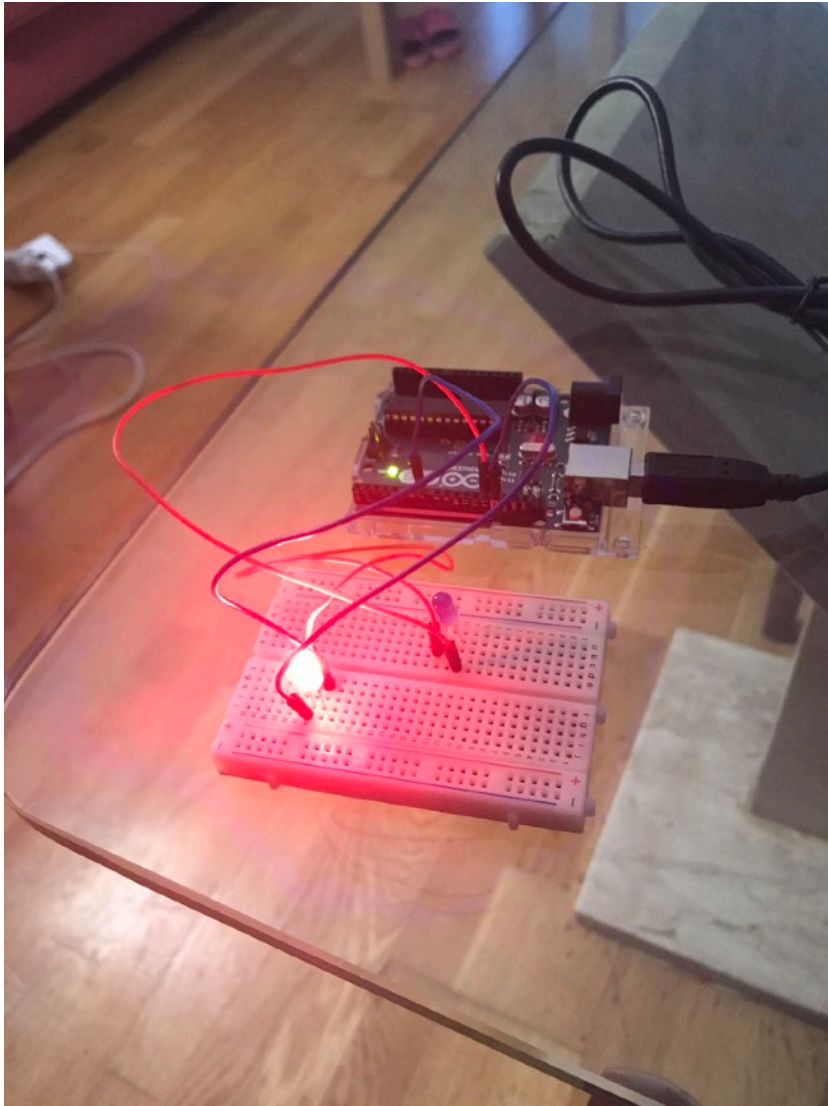
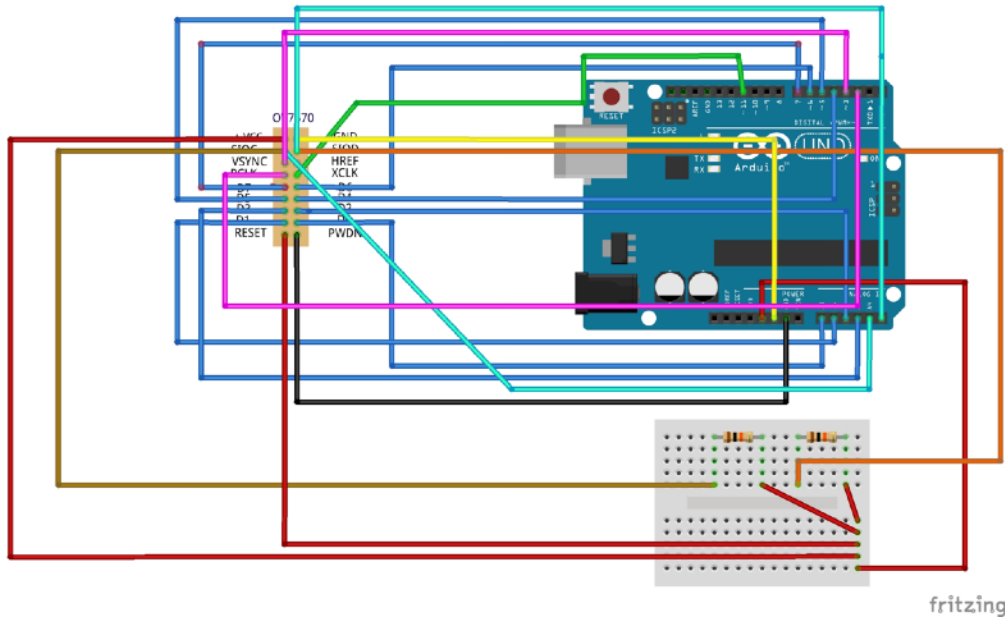


Figura 15. Desmostració de funcionament de concepte Arduino UNO

2.5 Implementació de la càmera OV7670

En aquest apartat es verificarà el funcionament del component OV7670 utilitzat en el projecte.

Figura 18. Esquema de muntatge càmera OV7670 amb Arduino UNO [8]



Es realitza el muntatge de la càmera seguint l'esquema proposat (fig. 18). I es procedeix al muntatge físic (fig. 19, fig. 20,).

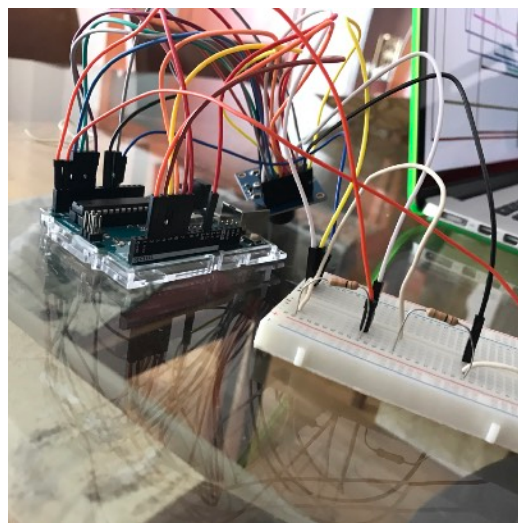


Figura 19. (1) Fotografia de muntatge de la càmera OV7670 amb Arduino UNO

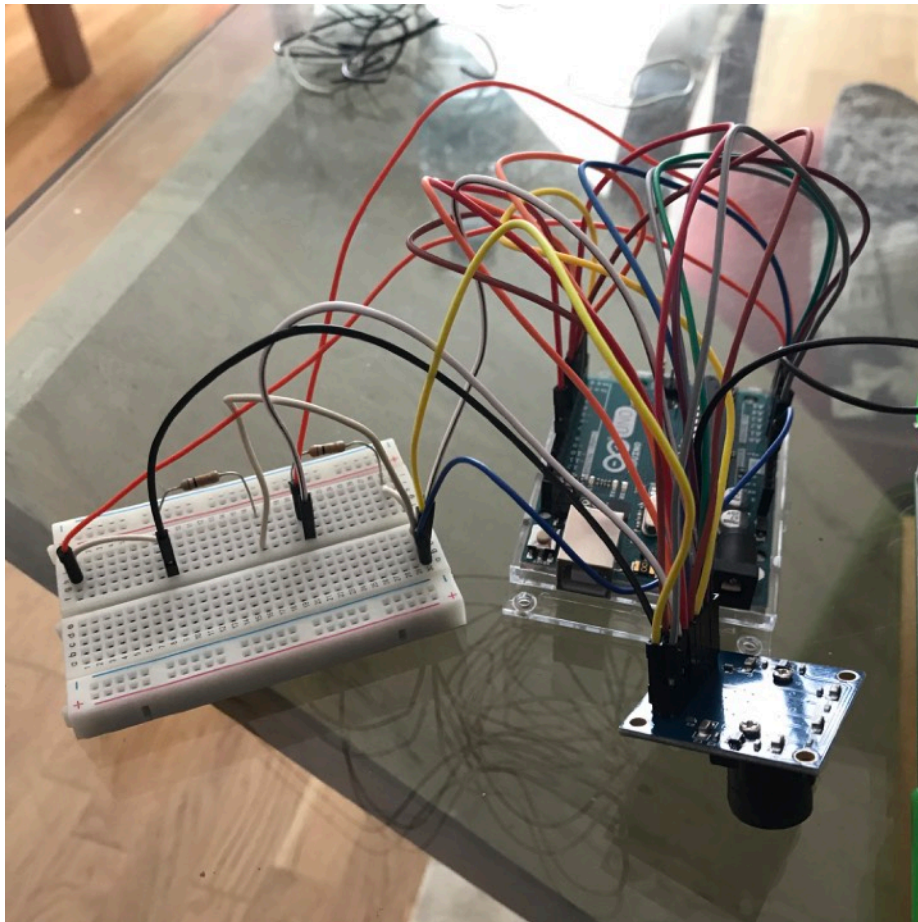


Figura 20. (2) Fotografia de muntatge de la càmera OV7670 amb Arduino UNO

Un cop acabada la part física, s'ha realitzat una recerca per a trobar la manera correcta de codificar la placa i que mostri imatges de la càmera.

En aquest punt s'han començat a trobar grans impediments per a fer funcionar la càmera OV7670 amb la placa Arduino UNO.

Buscant algun exemple per internet, solament existeix una compilació de codi [9] que suposadament funciona. A més, segons la descripció del propi autor, comenta que encara que s'aconsegueixi fer funcionar, sol donar diversos problemes de connectivitat, amb el que la fiabilitat, que és un punt molt important en el sistema que es vol implementar, es veu compromesa.

S'adjunta als annexos l'arxiu de codi de la càmera, ja que es un codi molt extens i difícil de comprendre.

A més, per a poder realitzar fotografies amb la càmera es necessita un programa extern que controli la informació que rep el canal per el qual es comunica Arduino amb l'ordinador, que es una entrada USB i capturar la imatge que es transmet donant-li la ordre de posar-se en marxa i de guardar una imatge.

Realitzant recerca per internet solament existeix un projecte que ho hagi realitzat i el codi no es accessible, amb el que no es viable la realització d'aquesta part, i ens estem allunyant seriosament de la construcció de la càmera i, per tant, de la realització del projecte.

El motiu d'aquesta complexitat en quant al muntatge i codi, és que el mòdul de la càmera no disposa d'un microcontrolador FIFO que organitza els píxels que es van creant després de la digitalització de la imatge presa per la lent. Llavors s'ha de construir manualment utilitzant els recursos de la placa. Però aquesta no està ideada per a processos amb tanta carrega de bits, llavors després quan volguessis dotar-li de connectivitat inal·làmbrica, aquesta no tindria suficients connexions i velocitat de processament com per a transmetre la imatge inal·làmbricament, amb el que resulta de molta complexitat poder realitzar aquesta combinació de Arduino UNO + Camera OV7670 + Wifi board CC3000. A més, el codi a compilar ocupa pràcticament tota la memòria de la placa, i encara quedaria implementar la connectivitat WiFi amb el mòdul d'Adafruit i el fet de que s'actives i avises a l'usuari quan detecta presència.

Vist aquesta situació, es torna a fer una valoració de la solució a implementar, i s'arriba a la conclusió que el sistema que es vol construir no reuneix els requisits que s'han definit per a aquest projecte. S'exposen les raons en el següent apartat:

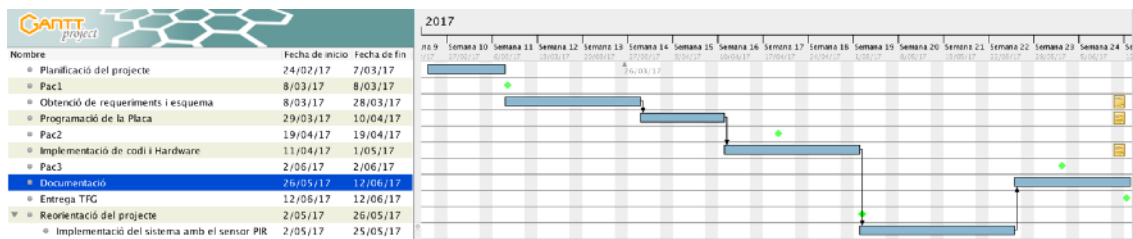
- No es fàcil d'implementar. El seu muntatge requeriria **més de 3 hores**.
- No es fiable. **El percentatge de fallades superaria el 10%** segurament.
- No es pot realitzar respectant el temps del qual es disposa en aquest projecte.
- No es pot afegir **compatibilitat inal·làmbrica** perquè Arduino UNO no té suficient memòria com per a emmagatzemar tot el codi que es necessita.
- No té suficients pins com per a connectar paral·lelament la placa WiFi, doncs pràcticament tots els pins estan ocupats, i el modul CC3000 n'ocupa bastants.
- No es té la certesa de que acabi funcionant dedicant-li el temps necessari.

Es conclou que el projecte, en aquestes condicions, **no és viable**.

3. Reorientació i elaboració del projecte

Un cop decidit el canvi d'estratègia, respectant l'objectiu del projecte i repassant els requeriments d'aquest en quant a la fabricació del producte, s'ha decidit realitzar la segona opció que s'havia proposat a la planificació del projecte. La **construcció d'un sensor de moviment que avisi a l'usuari quan detecta presència** de manera inal·làmbrica encara que aquest estigui fora de la seva llar.

Es realitza novament un diagrama de Gantt per a redefinir les tasques a nivell temporal:



Dins del marc de temps actual, el projecte podrà ser realitzable tot i haver rebut aquest contratemps. De fet, la tasca anomenada "implementació de codi i hardware" té una duració extensa precisament per a poder assolir canvis en el projecte, la qual cosa va ser descrita en l'apartat on figurava el diagrama de Gantt anterior.

3.1 Anàlisi de requeriments

El sistema continua sent poc costós, doncs un sensor de presència és més barat que una càmera OV7670 costant 1€, requereix de poques connexions i és fàcil d'implementar, amb el que resulta compatible amb la placa WiFi CC3000.

En aquest punt es planteja la manera en la que el sensor avisi a l'usuari quan es detecti moviment, i es decideix investigar una manera per a enviar a l'usuari un SMS amb un missatge dient-li que l'alarma s'ha activat. Actualment es coneixen pocs mètodes d'avís més eficaços que el de la recepció d'un SMS, doncs altres alternatives com l'enviament d'un correu electrònic o el fet d'haver de consultar l'estat de l'alarma en un servidor web, no són immediats, i restaria efectivitat al sistema.

D'aquesta manera, a més d'escollir una solució més senzilla, s'ha millorat la manera en que el sistema avisarà a l'usuari. I avui dia a majoria de persones té un telèfon mòbil capaç de rebre SMS.

Repasant l'anàlisi de requeriments del principi del projecte es compleixen tots:

- Immediat: El sensor de presència immediatament després de detectar un moviment avisarà a l'usuari mitjançant l'enviament d'un SMS. Es manté que **l'avís haurà d'arribar a l'usuari en menys de 2 minuts.**
- Fixe: El sensor de presència es situarà en un lloc fixe on monitoritzarà el moviment en tot moment. Encapsar el producte i fixar-lo **no requerirà de cap despesa econòmica** extra, utilitzant materials reciclats per a acomplir aquest requeriment.
- Connectat permanentment: el sistema pot funcionar permanentment, doncs tindrà un consum baix i estarà connectat a la xarxa WiFi. Es preveu que. **La connectivitat a internet i WiFi del punt d'accés haurà de ser 100% fiable** ja que sinó el sistema deixarà de funcionar i s'haurà de reiniciar manualment.
- Fiable: l'actuació del sistema es senzilla, detecta moviment, es connecta a internet i envia un SMS mitjançant algun servei web. És difícil un error en algun d'aquests punts descrits. Es preveu que el sistema mantindrà un **percentatge de fallades d'un 10%** degut al propi hardware i d'un **2% degut a problemes en quant a l'alimentació.**
- Fàcil de fer servir: l'usuari final solament haurà d'encendre el sensor de moviment i aquest començarà a treballar permanentment. Si es vol desactivar, solament s'ha d'apagar la placa. La seva configuració consistirà en modificar les seves dades personals..
- Fàcil d'instal·lar: El sistema solament s'ha d'instal·lar a una paret o sostre on el sensor de moviment apunti a l'àrea a detectar. També haurà d'estar connectat a la corrent mitjançant un cable.**El temps de construcció i implementació no hauria de sobrepassar les 3 hores**
- Inalàmbric: la connexió del sistema a internet es totalment inal·làmbrica, solament s'ha de connectar amb un cable a la corrent. **el sistema haurà d'estar permanentment connectat sense problemes a internet de manera indefinida.**

Un cop s'ha determinat que el sistema pot complir amb tots els requisits que exigeix es pot procedir amb la resta d'apartats.

3.2 Obtenció del sensor de moviment

El sensor adquirit es un PIR HC-SR501 (fig. 21).



Figura 21. Imatge d'internet d'un sensor PIR HC-SR501

Es un sensor de detecció de presència molt senzill i a la vegada funcional que envia una senyal quan es detecta qualsevol objecte que s'interposa en la seva àrea de detecció. Consta de dos reguladors per a ajustar la sensibilitat del sensor i té 3 pins de connexió que van cap a la placa Arduino. El seu cost ha sigut de 1€.

3.3 Implantació del sensor de moviment

Amb aquest component afegit al projecte al projecte, en comptes del mòdul de càmera, el nostre sistema quedarà de la següent forma:

- Es crearà un sistema de vigilància que consti d'un sensor de presència.
- Aquest sensor de presència enviarà un senyal a la placa quan detecti alguna intrusió en la zona on es col·loca.
- La placa Wi-Fi transmetrà via inal·làmbrica l'estat del sensor a un servidor domèstic.
- El servidor serà accessible via VPN per a qualsevol dispositiu que disposi de connectivitat i un navegador d'internet, introduint-hi unes credencials.
- El resultat serà, a nivell de cas d'ús, un usuari que podrà consultar en tot moment si hi ha presència a la zona on ha col·locat el dispositiu.
- Es valorarà implantar algun mètode d'avis a l'usuari quan el sistema d'alarma s'activi.

3.3.1 Muntatge de Hardware del sensor PIR

S'ha adquirit el sensor descrit anteriorment per 1,50€ (fig. 22).

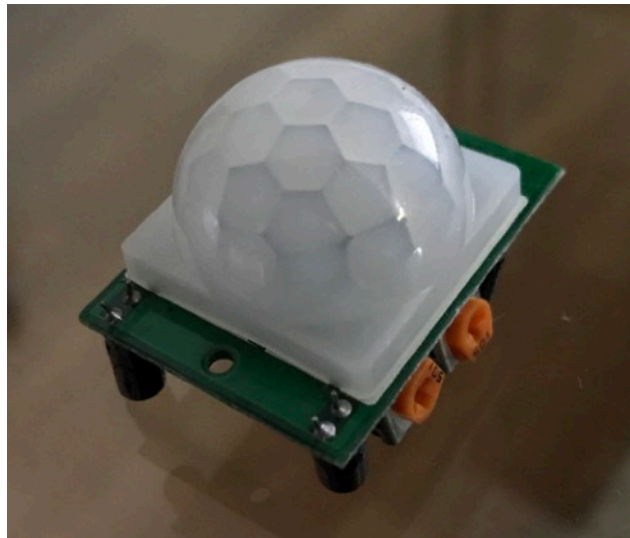


Figura 22. Fotografia d'un sensor PIR HC-SR501. Part frontal

El sensor consta de dos reguladors (fig. 22 i fig. 23) : un d'ells serveix per a regular la sensibilitat del sensor i l'altre per a establir el temps que triga a tornar a activar-se per a realitzar un altre cop la següent detecció i enviament del senyal si detecta moviment.



Figura 23. Fotografia d'un sensor PIR HC-SR501. Part darrera

I tal com s'ha previst en l'apartat anterior es procedeix a realitzar una prova de concepte amb la protoboard utilitzada per als anteriors components. S'ha realitzat un muntatge que consta de el Sensor PIR, la placa Arduino UNO, la protoboard i dos LEDS: un verd i l'altre vermell. L'objectiu és que quan el sensor PIR detecti presència, s'activi el LED vermell. I mentre no detecti res ,es mantingui activat el LED verd. S'ha dissenyat un esquema per a obtenir el muntatge de hardware (fig. 28).

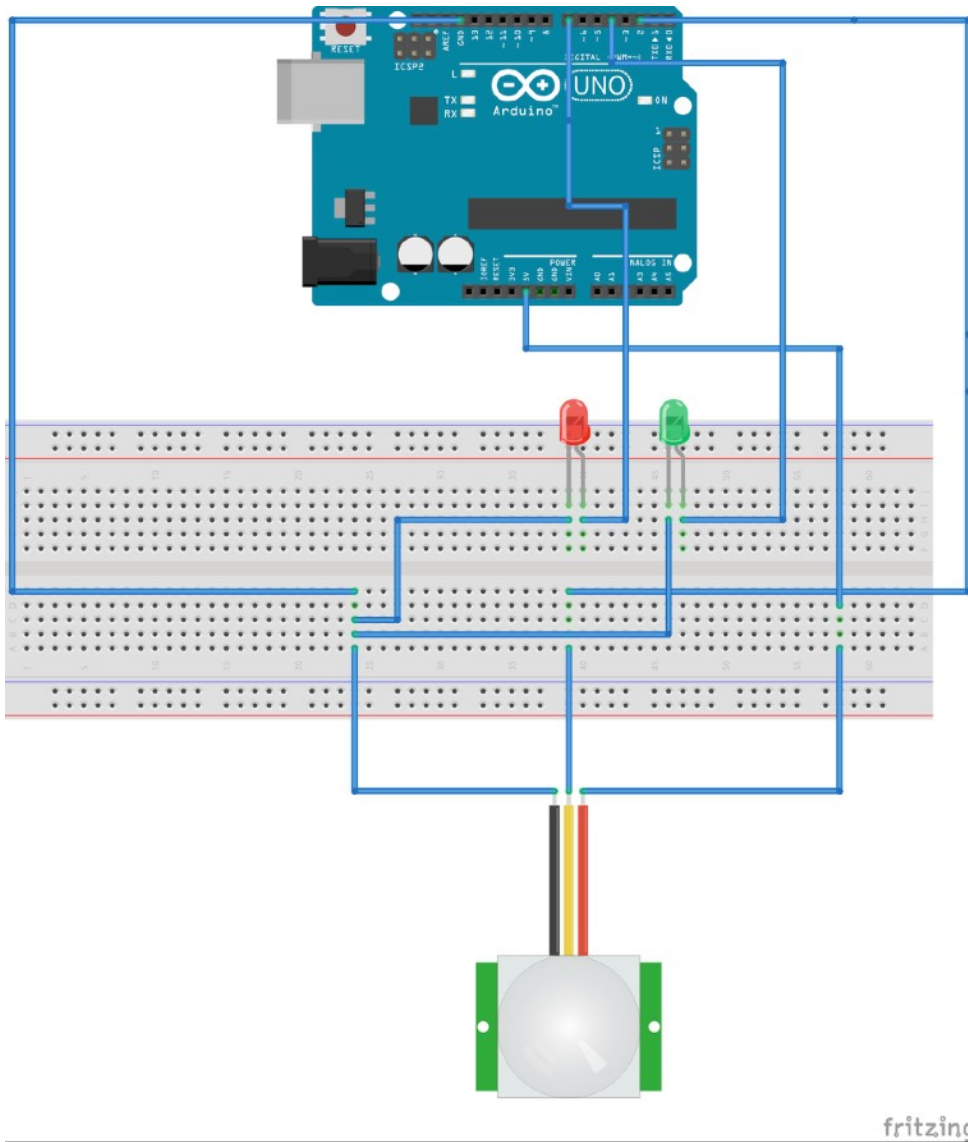


Figura 24. Esquema de muntatge de la prova de concepte amb el sensor PIR

El muntatge no és complex i ocupa poques entrades/sortides de la placa Arduino (fig. 25).

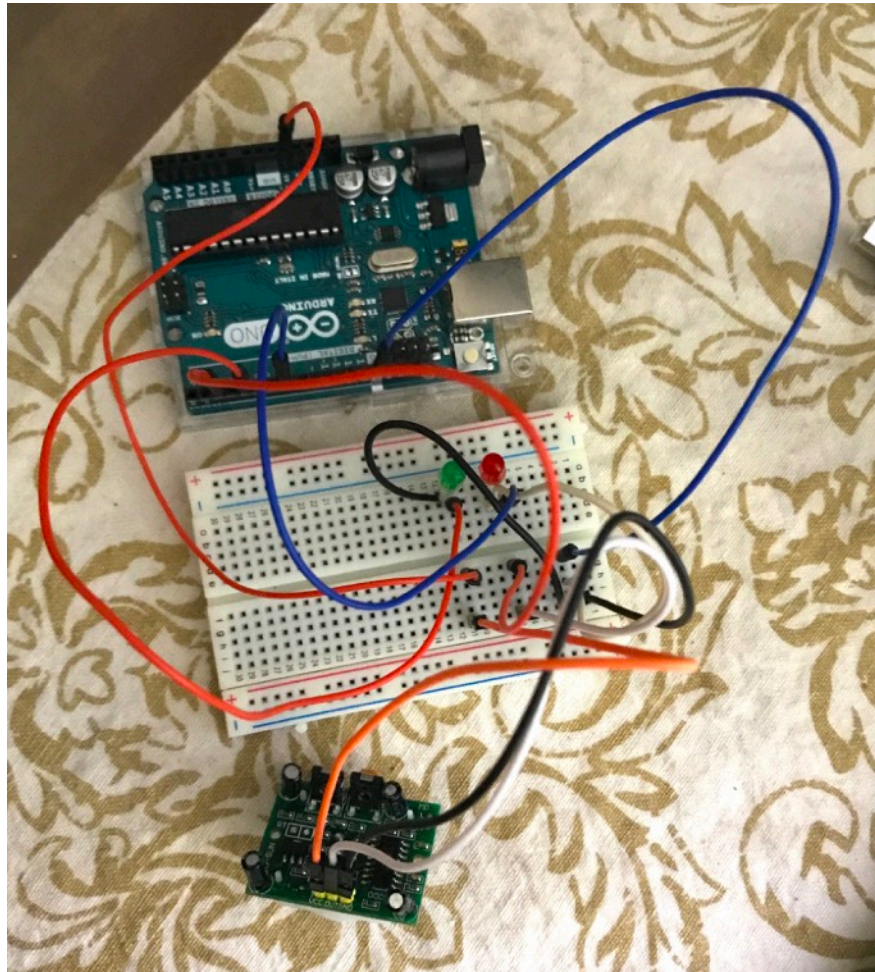


Figura 25. Fotografia del muntatge físic de la prova de concepte del sensor PIR

- El sensor de moviment rep 5 volts d'Arduino UNO(sortida de la placa de 5V) per a la seva activació, i l'enviament del senyal de presència va connectat a l'entrada digital numero 2 de la placa.
- Els LEDs va connectats de la següent manera: el verd a la sortida 4 i el vermell a la sortida 7. Totes les sortides son digitals.
- Es poden apreciar que tant els pins com el sensor de moviment queden connectats a la mateixa presa de terra.

3.3.2 Programació del sensor PIR

El codi de programació de la placa utilitzat compleix les condicions per a funcionar amb el descrit a l'apartat anterior.

```
//variables definition
const int RedPin= 7;
const int GreenPin = 4;
const int SensorPin = 2;

void setup()
{
  //pin modes
  Serial.begin(9600);

  pinMode (RedPin, OUTPUT);
  pinMode (GreenPin, OUTPUT);
  pinMode (SensorPin, INPUT);
}

void loop() {
  // put your main code here, to run repeatedly:
  int detector = digitalRead(SensorPin);

  //detection condition
  if (detector == HIGH)
  {
    //if somebody is detected
    digitalWrite(RedPin,HIGH);
    digitalWrite(GreenPin, LOW);
    Serial.println("Detectado!");
  }

  else
  {
    //if nobody is detected
    digitalWrite(GreenPin, HIGH);
    digitalWrite(RedPin, LOW);
    Serial.println("NO Detectado!");
  }
}
```

Codi 7. Prova de concepte del sensor PIR

3.3.3 Prova de funcionament del sensor PIR

Un cop verificat el codi, compilat i carregat a la placa, podem verificar que el seu funcionament es l'esperat (fig. 26).

Quan no es detecta moviment:

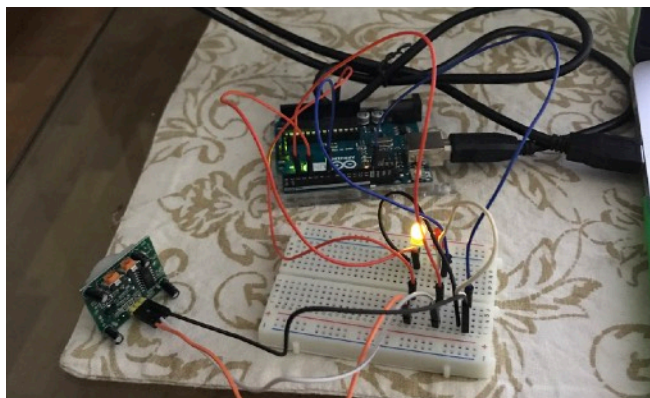


Figura 26. Demostració de la prova de concepte del sensor PIR quan no detecta moviment

Quan el sensor detecta moviment el led verd s'apaga i s'encén el verd (fig. 27).

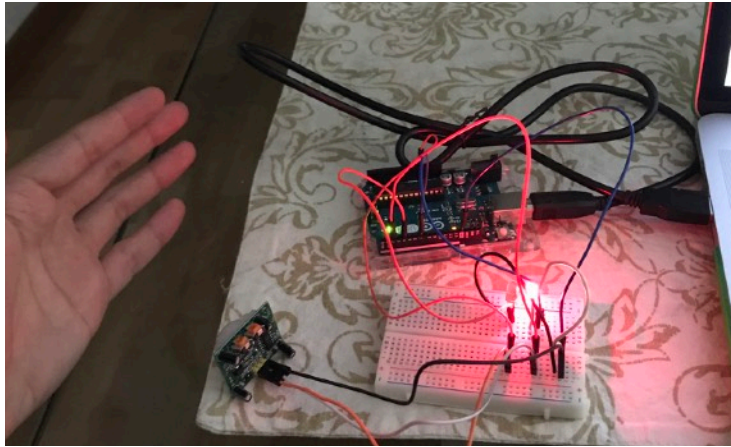


Figura 27. Demostració de la prova de concepte del sensor PIR quan detecta que hi ha moviment

Amb la realització de la prova de concepte, s'ha obtingut un sensor de moviment preparat per a enviar un senyal quan es detecti presència. A partir d'aquí, es podran enviar ordres a la placa com en aquesta demostració.

3.4 Estratègia d'implantació WiFi

L'objectiu de la implementació del projecte és connectar tot el sistema amb un a xarxa Wi-Fi. Si s'utilitza connectivitat Ethernet per a realitzar la connexió a internet, llavors el sistema haurà d'estar prop del router i es restringeix bastant la localització del sistema.

Per mitigar aquesta dificultat, s'ha decidit, tal com es va plantejar al principi del projecte, l'ús d'una placa Wi-Fi compatible amb Arduino UNO, la qual dotarà el nostre sistema de connectivitat inal·làmbrica per a poder informar a l'usuari del canvi d'estat del sensor PIR en qualsevol lloc.

La placa ja ha estat explicada en els anteriors apartats, amb el que es procedeix directament a realitzar el muntatge d'aquesta per a després proposar una solució de codi.

3.4.1 Muntatge de Hardware del mòdul Adafruit CC3000

Primerament s'ha d'observar que la placa CC3000 no té pins de connexió com la resta de components, amb el que hem d'utilitzar un petit component (fig. 28) que, soldat a la placa wifi, aquesta queda preparada per a poder connectar-se fàcilment a una protoboard:

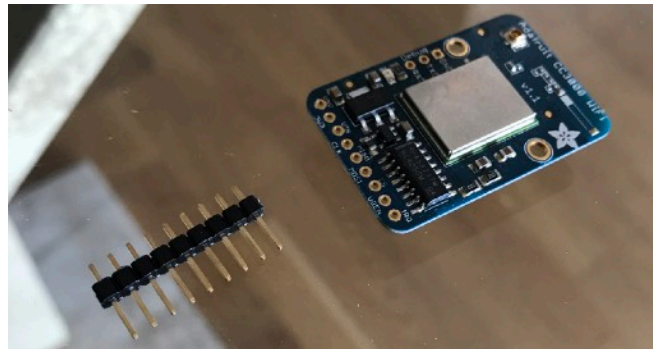


Figura 28. Fotografia del component WiFi Adafruit CC3000

El procediment consisteix en introduir els pins del petit component per els forats de la placa Wi-Fi, que són els connectors que aniran directament a la protoboard, i d'aquí a la placa Arduino UNO (fig. 29).

Finalment ens proposem connectar, mitjançant la protoboard la placa Wi-Fi i, a més, connectar-la al nostre sensor de moviment. Per a això s'han d'estudiar detingudament els seus connectors.

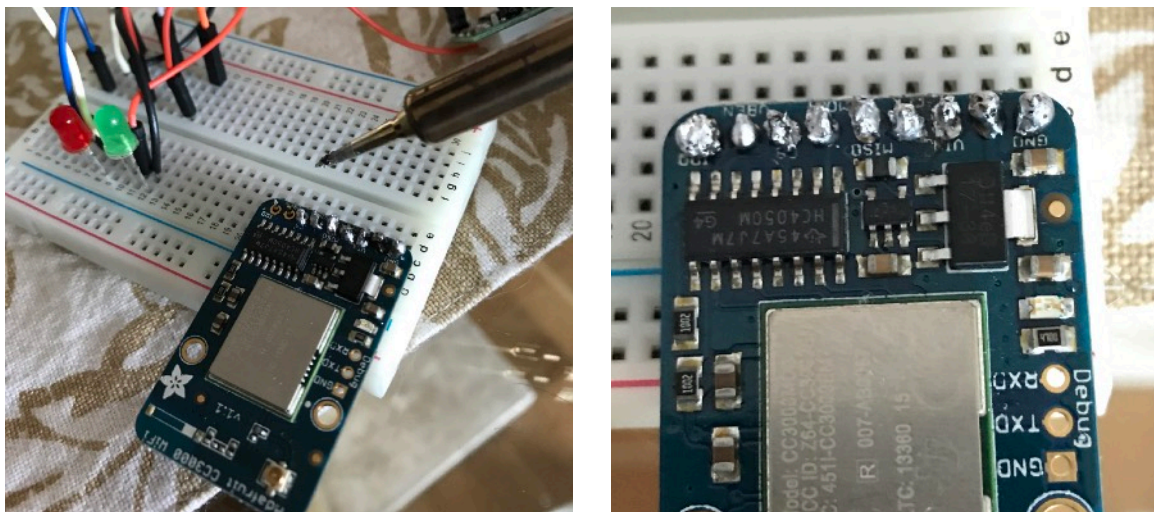


Figura 29. a) Fotografia del component WiFi Adafruit CC3000 soldat a un array de connectors i introduït a una protoboard CC3000. b) Fotografia ampliada del component Adafruit CC3000 connectat a una protoboard.

La pròpia pàgina del fabricant Adafruit ens proveeix de tota la documentació necessària respecte el funcionament de la placa[12].

Aquesta pàgina web ens indica l'esquema de muntatge adequat per a aquest component amb Arduino(fig. 30).

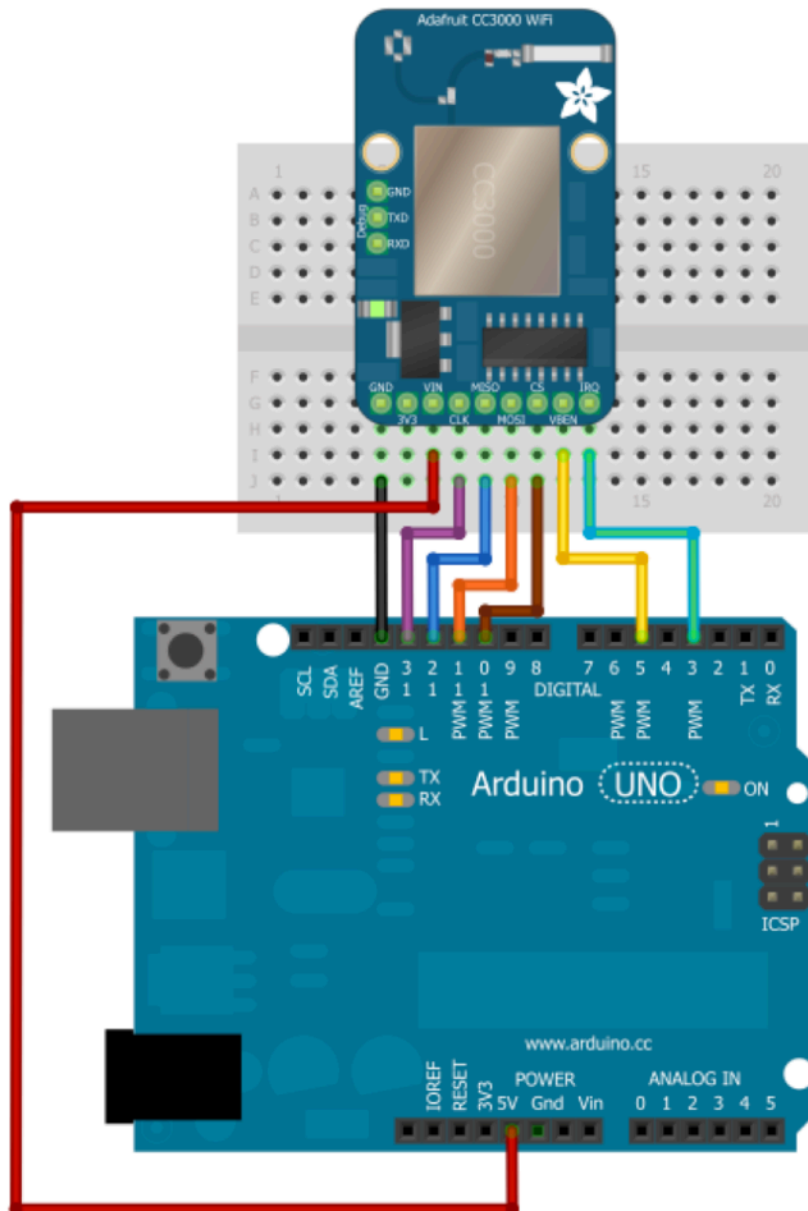


Figura 30. Imatge d'un esquema de muntatge del component Adafruit CC3000 amb Arduino UNO utilitzant una protoboard [12]

En quant a l'ús de cada pin de la placa Wi-Fi, trobem un fragment amb la seva explicació de cadascun a l'enllaç anterior del tutorial de Adafruit[12]:

“

The CC3000 is (electrically) fairly simple to use. The module requires an SPI connection, including a clock (CLK), data in from a microcontroller (MOSI) and data out to the microcontroller (MISO). It also uses a chip-select line (CS) for SPI to indicate when a data transfer as started

Along with the SPI interface, there is a power-enable type pin called VBAT_EN which we use to start the module properly and also an IRQ pin, which is the interrupt from the CC3000. The IRQ pin is required to communicate and must be tied to an interrupt-in pin on the Arduino. On the Mega/UNO, we suggest #2 or #3

”

El que detalla aquest fragment, es la funció principal de cada pin de la placa inal·làmbrica. Es pot extreure que aquesta consta d'una connexió SPI (Serial Peripheral Interface), que consta d'un connector CLK per a sincronitzar correctament la freqüència de les dades trameses, dos connectors amb microcontrolador de dades tant d'entrada com de sortida: MOSI i MISO, i un connector CS (Chip Select) per a sincronitzar quan ha començat la transmissió de dades.

A part dels connectors SPI, tenim diversos pins per a acabar de fer funcionar el sistema sencer: un connector anomenat VBAT_EN que s'utilitzar per a encendre el mòdul, i l'interruptor d'aquest que es l'entrada IRQ.

A continuació, es realitzarà una prova de concepte a comprovar el funcionament de la placa Wi-Fi amb Arduino UNO i testejar la seva funcionalitat.

Es crea un esquema de fabricació per al component CC3000, però s'aprofitarà l'anterior esquema de muntatge amb el sensor PIR utilitzant els connectors que han quedat lliures. D'aquesta manera es podrà implementar directament un sistema final de hardware.

Es crea un esquema de muntatge que serà el definitiu en quant a la part de hardware (fig. 31).

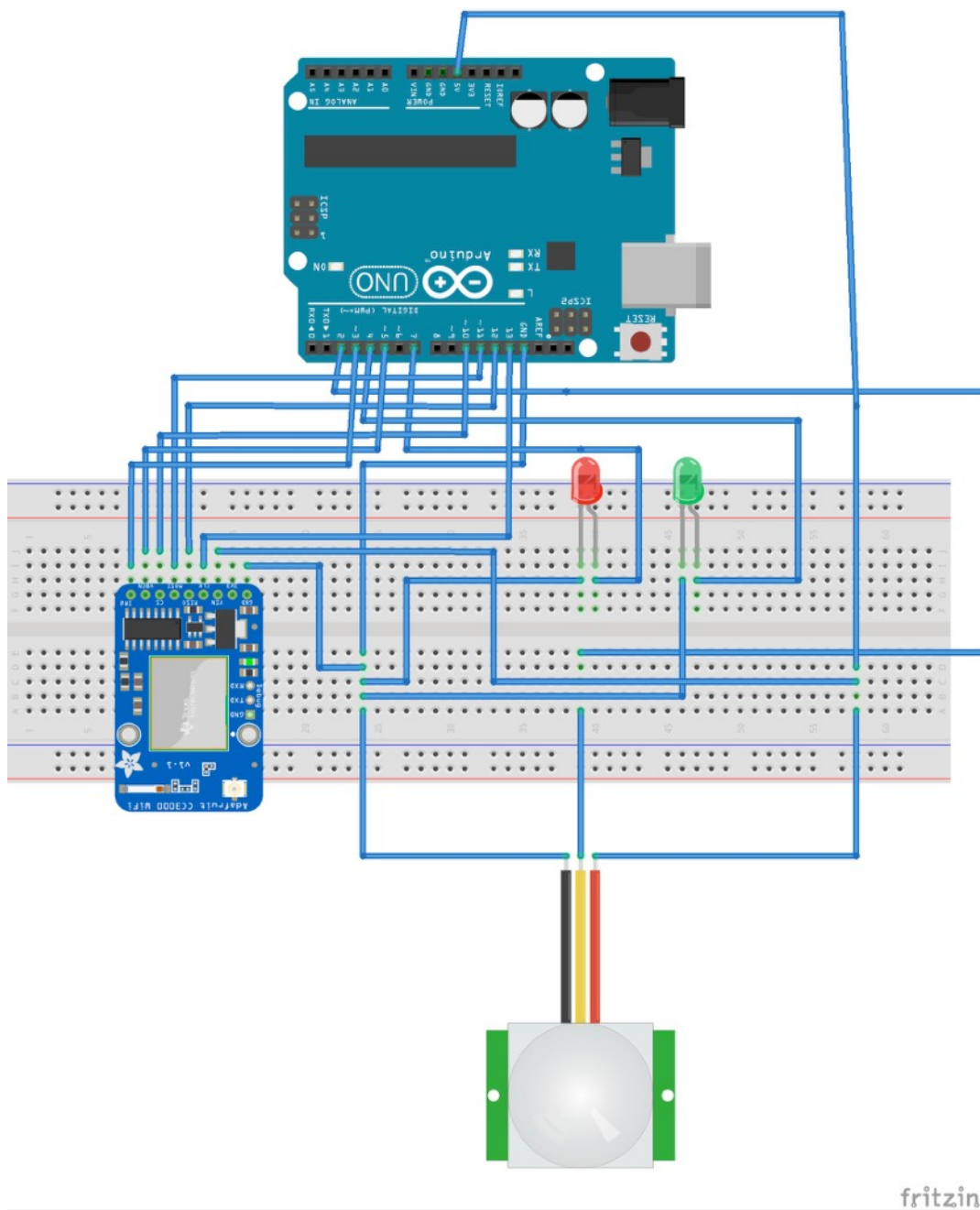


Figura 31. Esquema de muntatge realitzat utilitzant el sensor PIR, Arduino UNO i el mòdul Wifi Adafruit CC3000 per a crear el sistema final del projecte

Es connecten els components tal com es mostra a l'esquema de muntatge(fig. 38) respectant les connexions ja realitzades, que son referents al sensor PIR. Es construeix el sistema final a nivell de hardware (fig. 32).

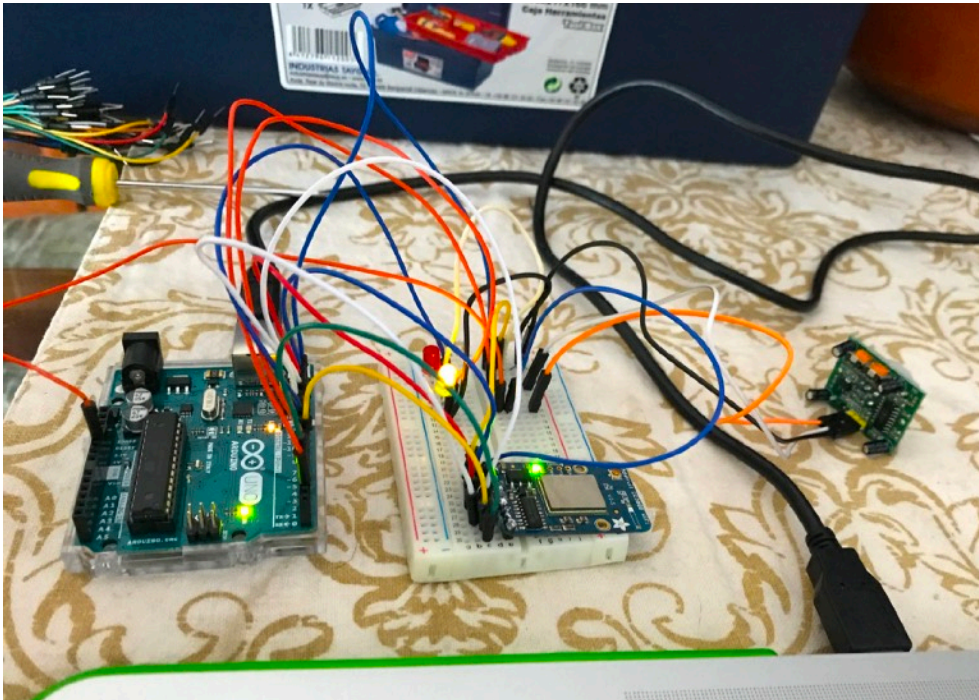


Figura 32. Imatge del sistema final construït (Hardware)

Finalment es posarà a prova la seva funcionalitat amb un petit test que comprovarà la connectivitat de la placa WiFi a una xarxa domèstica.

Per a programar la placa, necessitem diverses llibreries que el propi fabricant Adafruit distribueix de forma lliure[17]. [Annex I.2]

I el seu Schematic es troba al annexos [Annex I.6].

3.4.2 Test de la placa Adafruit C3000

Es farà servir la llibreria d'exemple anomenada "built code", que trobem en les llibreries distribuïdes per Adafruit (fig. 33).



```
buildtest Arduino 1.8.2
buildtest
* Disconnect
It's a good idea to run this sketch when first setting up the
module.

*/

#include <Adafruit_CC3000.h>
#include <ccspi.h>
#include <SPI.h>
#include <string.h>
#include "utility/debug.h"

// These are the interrupt and control pins
#define ADAFRUIT_CC3000_IRQ 3 // MUST be an interrupt pin!
// These can be any two pins
#define ADAFRUIT_CC3000_VBAT 5
#define ADAFRUIT_CC3000_CS 10
// Use hardware SPI for the remaining pins
// On an UNO, SCK = 13, MISO = 12, and MOSI = 11
Adafruit_CC3000 cc3000 = Adafruit_CC3000(ADAFRUIT_CC3000_CS, ADAFRUIT_CC3000_IRQ, ADAFRUIT_CC3000_VBAT,
SPI_CLOCK_DIVIDER); // you can change this clock speed but DI

#define WLAN_SSID "Toby 2.4G" // cannot be longer than 32 characters!
#define WLAN_PASS "Nuria1990"
// Security can be WLAN_SEC_UNSEC, WLAN_SEC_WEP, WLAN_SEC_WPA or WLAN_SEC_WPA2
#define WLAN_SECURITY WLAN_SEC_WPA2

/*****
/*!
 @brief Sets up the HW and the CC3000 module (called automatically
 on startup)
 */
*****/
```

Figura 33. Sketch que mostra el codi implementat de la connectivitat amb una placa Arduino UNO i el component Adafruit CC3000

Aquest Sketch el que fa es realitzar una prova de connexió de la placa Adafruit C3000 amb una xarxa WiFi coneguda i propera. També comprova que la placa s'encengui i sigui capaç de trobar xarxes properes. Si s'aconsegueix connectar amb la nostra xarxa WiFi s'imprimirà per un monitor serial la xarxa com la IP, Gateway, Netmask i alguns camps més (fig. 42). El monitor descrit es un component que conté el software Arduino IDE.

```

/dev/cu.usbmodem1411 (Arduino/Genuino Uno)
Hello, CC3000!

RX Buffer : 131 bytes
TX Buffer : 131 bytes
Free RAM: 1171

Initialising the CC3000 ...
Firmware V. : 1.24
MAC Address : 0x08 0x00 0x28 0x58 0xA0 0x82
Networks found: 1
=====
SSID Name    : Toby 2.4G
RSSI         : 45
Security Mode: 3
=====

Deleting old connection profiles

Attempting to connect to Toby 2.4G
Connected!
Request DHCP

IP Addr: 192.168.1.139
Netmask: 255.255.255.0
Gateway: 192.168.1.1
DHCPsrv: 192.168.1.1
DNSServ: 192.168.1.1
www.adafruit.com -> 104.20.39.240
Pinging 104.20.39.240...5 replies
Ping successful!

Closing the connection

[✓] Autoscroll      Sin ajuste de línea  115200 baudio

```

Figura 34. Captura de la sortida Serial d'Arduino UNO amb el Sketch "built" utilitzant el component Adafruit CC3000

Modifiquem el codi per a afegir-li les credencials de la nostra xarxa tal com s'ha mostrat a la imatge anterior i obtenim la següent sortida de dades:

Amb això queda demostrat el correcte funcionament del component Adafruit CC3000, ja que es connecta a la nostra xarxa, ens mostra informació d'aquesta i realitza una prova de connexió (ping) amb èxit (fig. 34).

4. Implementació del sistema

Un cop s'ha comprovat la funcionalitat de tots els components que participen en el sistema i correctament connectats a la protoboard, en primer lloc s'intentarà crear un sistema per a consultar el sensor de moviment a distància. A més, per a que l'objectiu del projecte sigui realment funcional, es necessita un mecanisme que avisi a l'usuari d'aquesta consulta quan l'alarma s'activa.

4.1 Anàlisi de l'arquitectura del sistema a implementar

Investigant diverses maneres d'implementar la funcionalitat d'avís, s'ha pensat en la simplicitat i eficiència que ofereix l'ús del SMS com a mètode d'enviament del missatge que informi a l'usuari que l'alarma s'ha activat. Aquest tipus d'avís sol ser immediat i simple.

l mirant diverses implementacions, resulta que la placa WiFi té una llibreria que ofereix serveis ofert per Temboo, una empresa que ofereix serveis Open Source per a dispositius IoT (Internet of Things). Un cop donat d'alta, de manera gratuïta, la pàgina ens mostra diverses opcions i serveis.

Temboo el que fa es crear codi sketch de forma automàtica per Arduino, indicant-li quina funció es vol realitzar, quin model d'Arduino es té i quina connectivitat (hardware) s'utilitza. A partir d'aquí, Temboo ofereix diverses solucions anomenades "choreos", que no son més que llibreries que integren la solució i l'adapten segons les dades introduïdes anteriorment. En aquest cas, es vol una solució de connectivitat d'enviament de SMS el qual s'activi amb una condició concreta, que per a nosaltres es l'activació del sensor.

l es en aquest punt on s'utilitza un altre servei que és compatible amb Temboo. és un servidor web anomenat Twilio[15], i ofereix un servei gratuït per a enviar SMS a qualsevol telèfon de manera automatitzada. Temboo genera codi sketch el qual utilitza els serveis de Twilio per a realitzar aquesta acció.

Twilio envia SMS de manera gratuïta des de un número mòbil que et proporciona ell mateix. Un cop donat d'alta pagina, ofereix diversos serveis, entre els quals es té el desitjat en aquest projecte, l'enviament automatitzat de SMS (fig. 35).

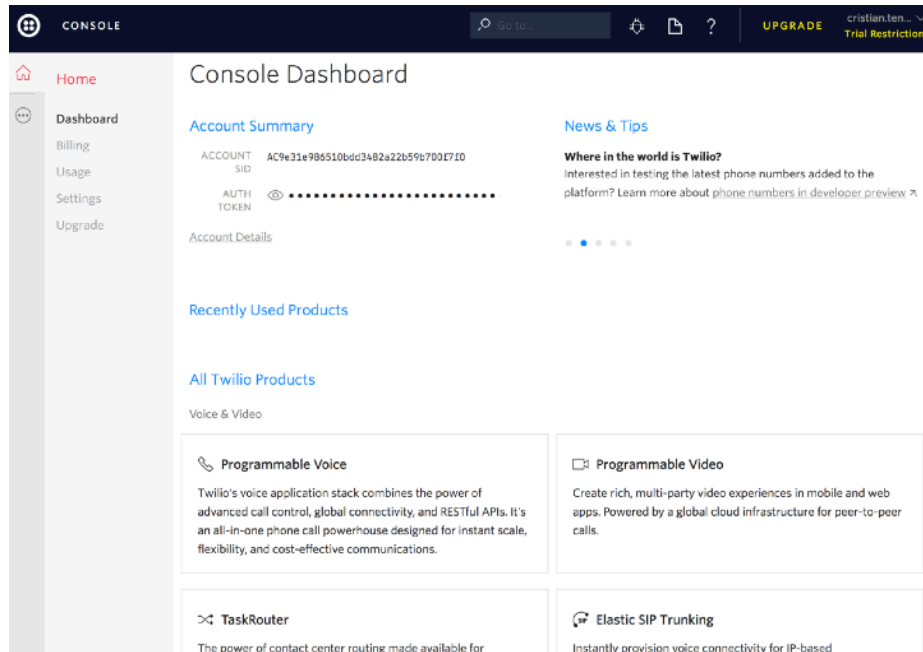


Figura 35. Portal principal d'usuari de la pàgina web de Twilio [15]

Es crea un esquema que mostra gràficament l'arquitectura que tindrà el sistema final que es vol implementar (fig. 36).

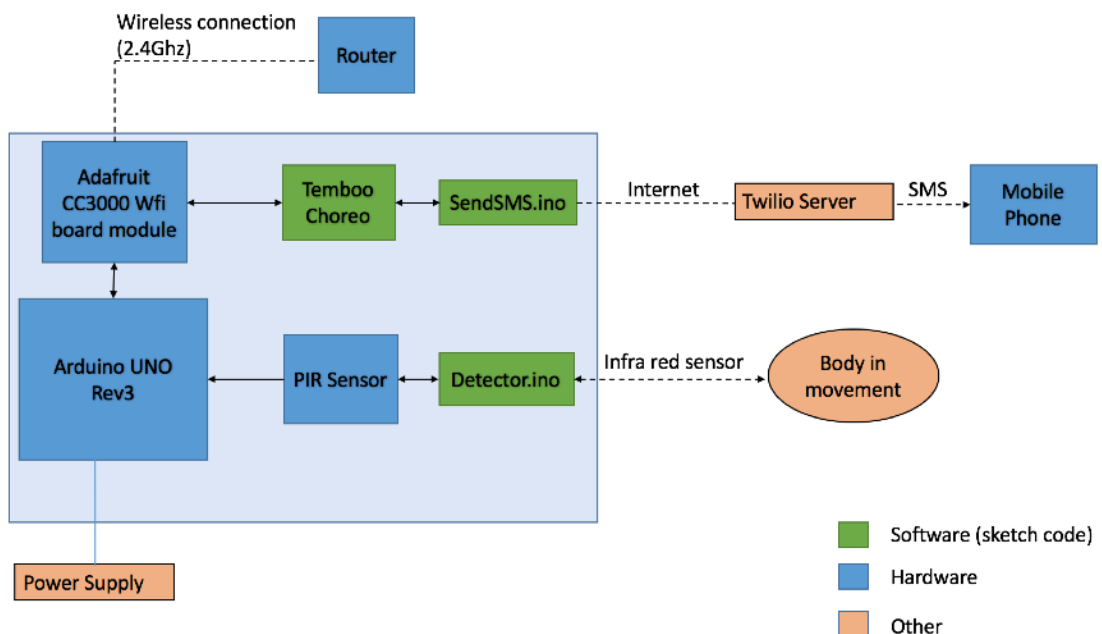


Figura 36. Esquema conceptual de l'arquitectura del sistema a implementar

4.1.1 Anàlisi de codi del sketch de Temboo

Per a comprovar el funcionament de la llibreria Temboo[14], el primer que es realitzarà serà una prova de concepte que demostrï el funcionament de manera simplificada d'aquesta llibreria. Es seguirà el tutorial de la pròpia pàgina per a entendre el seu funcionament[13].

Es procedeix a connectar i carregar codi des de Temboo a Arduino i fer-lo funcionar. Arduino UNO porta inclosa la llibreria Temboo, la qual cosa facilita la feina, ja que carregar-la es relativament fàcil.

El tutorial ens ensenya a crear un petit servei de geolocalització que transforma una direcció postal que li passem a coordenades polars. En el nostre cas, ho programarem per a que quan el sensor s'activi, enviarà un senyal a la sortida digital numero 8, i llavors es dispararà el codi que generarà la conversió de geolocalització. La sortida es pot indicar de manera gràfica i, a més, pots indicar l'interval de temps que ha de passar per a que s'activi el codi de Temboo.

I es en aquest punt on es troba la major dificultat del projecte a nivell d'implementació de codi.

Resulta que Temboo no dona suport per a la placa WiFi que s'està fent servir en aquest projecte, Adafruit CC3000. Solament dona recursos de generació de codi per a plaques basades en Arduino Yun, ja que aquesta porta implementades unes llibreries que donen molta facilitat a l'hora d'implementar una connectivitat ja sigui inal·làmbrica o per Ethernet.

El codi mostrat a continuació és el generat amb la prova de concepte utilitzant Arduino UNO i la placa Arduino Yun Shield. Es mostra un fragment important de codi a analitzar. El codi complet es troba als annexos [annex I.6]:

```
/*  
  GetYahooWeatherReport  
  
  Demonstrates making a request to the Yahoo Weather API using the Temboo Arduino Yun library.  
  
  This example code is in the public domain.  
*/  
  
#include <Bridge.h>  
#include <Temboo.h>  
#include "TembooAccount.h" // contains Temboo account information  
...
```

Codi 2. Exemple d'iniciació de Temboo

Si ens fixem en l'apartat del void setup(), que és on es crea la configuració de connectivitat, solament tenim una funció que mostra enllaç amb internet:

```
void setup() {  
  Serial.begin(9600);  
  
  // for debugging, wait until a serial console is connected  
  delay(4000);  
  while(!Serial);  
  Bridge.begin();  
}
```

La funció **Bridge.begin()** automàticament estableix connectivitat a internet, cosa que resulta molt senzilla, però no són les llibreries que utilitza Adafruit CC3000. Són les de Arduino Yun (una altre model de d'Arduino).

Es conclou que Temboo no es capaç de crear automàticament un codi adaptat per al mòdul Wi-Fi Adafruit CC3000. Ja que les llibreries que incorporen els models Yun son completament diferents a les que es necessiten, amb el que el Sketch que es crea automàticament per Temboo no es vàlid per a el nostre hardware.

Arribat a aquest punt, s'estableix com a objectiu principal d'aquest apartat crear un sketch capaç de compatibilitzar el mòdul WiFi Adafruit CC3000 amb el codi generat per Temboo (SendSMS). Adafruit implementa la seva pròpia connectivitat, i si bé el codi de Temboo no és valid per a que la placa es connecti, la part de codi dedicada a la resta de l'aplicació(SendSMS) si que ho és.

Per a utilitzar aquesta llibreria, sense entrar a nivell de codi, s'han de realitzar les següents passes per a generar un sketch compatible que més endavant intentarem modificar per a adaptar-lo al mòdul WiFi CC3000.

S'obtenen les credencials necessàries i es genera un sketch per a treballar-hi com a base. A l'annex [annex I.5] es troben les passes necessàries per a poder accedir al codi SendSMS de Twilio.

4.2 Implementació de la solució de codi

Per a poder implementar el codi de SendSMS i el codi de de la nostre placa WiFi, s'han d'anal·litzar cadascun d'ells, entendre quines funcions realitzen i crear un sketch funcional que utilitzi la funció SendSMS mitjançant la connectivitat de la placa CC3000.

4.2.1 Anàlisi del Sketch d'Adafruit CC3000

S'analitza el codi utilitzat per a la placa CC3000.

```
#include <Adafruit_CC3000.h>
#include <ccspi.h>
#include <SPI.h>
#include <string.h>
#include "utility/debug.h"
#define ADAFRUIT_CC3000_IRQ 3 // MUST be an interrupt pin!
#define ADAFRUIT_CC3000_VBAT 5
#define ADAFRUIT_CC3000_CS 10
Adafruit_CC3000 cc3000 = Adafruit_CC3000(ADAFRUIT_CC3000_CS, ADAFRUIT_CC3000_IRQ,
ADAFRUIT_CC3000_VBAT,
SPI_CLOCK_DIVIDER);

#define WLAN_SSID "SSIDXarxa Wifi" // cannot be longer than 32 characters!
#define WLAN_PASS "*****"
#define WLAN_SECURITY WLAN_SEC_WPA2
void setup(void)
{
  Serial.begin(115200);

  if (!cc3000.begin())
  {
    Serial.println(F("Unable to initialise the CC3000! Check your wiring?"));
    while(1);
  }

  if (!cc3000.setDHCP()) {
    Serial.println(F("Failed to set DHCP!"));
    while(1);
  }
  if (!cc3000.connectToAP(WLAN_SSID, WLAN_PASS, WLAN_SECURITY)) {
    Serial.println(F("Failed!"));
    while(1);
  }

  Serial.println(F("Connected!"));

  while (!cc3000.checkDHCP())
  {
    delay(100); // ToDo: Insert a DHCP timeout!
  }
  /* Display the IP address DNS, Gateway, etc. */
  while (!displayConnectionDetails()) {
    delay(1000);
  }
  cc3000.disconnect();
}
```

Codi 3. Sketch BuiltTest Adafruit CC3000

A continuació es mostren les parts principals del codi anterior [Codi 3]:

- Lliberies Adafruit:
 - #define ADAFRUIT_CC3000_IRQ 3
 - #define ADAFRUIT_CC3000_VBAT 5
 - #define ADAFRUIT_CC3000_CS 10

Aquestes llibreries són les necessàries per a que el codi Sketch hereti les funcions utilitzades posteriorment al codi. S'està definint a quin connector de la placa Arduino UNO es realitza cada funció definida.

- Instància de connectivitat Adafruit CC3000:
 - `Adafruit_CC3000 cc3000 =
Adafruit_CC3000(ADAFRUIT_CC3000_CS,
ADAFRUIT_CC3000_IRQ, ADAFRUIT_CC3000_VBAT,`

Es genera una instància requerint els tres paràmetres definits mitjançant les llibreries anteriors. Aquesta instància anomenada en el nostre cas "cc3000" conté els valors necessaris per a enviar l'informació procedent dels pins procedents de la placa CC3000 cap a qualsevol codi que demani connexió a internet.

Funcions de la instància generada "cc3000":

- `cc3000.begin`: inicialitza la placa WiFi i realitza connexió a internet agafant les credencials següents:
 - `#define WLAN_SSID`: nom de la xarxa WiFi a la que ens volem connectar
 - `#define WLAN_PASS`: Contrasenya de la xarxa WiFi
 - `#define WLAN_SECURITY`: tipus de seguretat que utilitza el punt d'accés.
- `cc3000.setDHCP`: es crea una connexió DHCP en la qual se li assigna a la placa una direcció IP per a poder connectar-se.
- `cc3000.connectToAP`: són els acrònims de "connect to access point" i ens retorna el resultat d'aconseguir connectar-se al punt d'accés definit mitjançant les nostres credencials.
- `cc3000.checkDHCP`: retorna una variable 'true' o 'false' depenent si el dispositiu està correctament connectat o no a la xarxa WiFi.
- `cc3000.disconnect`: finalitza tots els processos de la placa WiFi.

L'anàlisi de codi i el procés d'enteniment d'aquest sketch es pot visualitzar amb més detall als annexos [annex I.3].

4.2.2 Anàlisi del codi del “Chorus” SendSMS de Temboo

Un cop analitzat el codi de la placa WiFi, es realitza el mateix procediment per a trobar un enllaç de connectivitat entre el Sketch SendSMS i del d’Adafruit.

En el codi per defecte generat per Temboo, el chorus SendSMS, es pot observar que per a instanciar una connectivitat a internet es fa servir la següent funció:

- `TembooChoreo SendSMSChoreo(client);` : es pot observar com es genera una instància anomenada `SendSMSChoreo` i que se li passa el paràmetre “client”. Aquest paràmetre no son més les credencials obtingudes per a establir connexió a internet.
- `SendSMSChoreo.begin();` : es pot apreciar que aquesta instància realitza algunes de les funcions de connectivitat que les de la placa WiFi CC3000.

S’observa doncs, que proporcionant-li com a paràmetre de connectivitat la instància generada per a la placa CC3000, es pot generar una altra nova instància amb Temboo que utilitza les mateixes variables.

Es crea a continuació un sketch que comprova que la connectivitat entre la placa WiFi i la funció SendSMS és correcte mitjançant els mètodes estudiats anteriorment.

```

#include <SPI.h>
#include <Adafruit_CC3000.h>
#include <Adafruit_CC3000_Server.h>
#include <ccspi.h>
#include <Client.h>
#include <Temboo.h>
#include "TembooAccount.h"

#define TEMBOO_APP_KEY_NAME "myFirstApp"
#define ADAFRUIT_CC3000_IRQ 3
#define ADAFRUIT_CC3000_VBAT 5
#define ADAFRUIT_CC3000_CS 10

#define WIFI_SSID "Toby 2.4G" // cannot be longer than 32 characters!
#define WPA_PASSWORD "*****"
#define WLAN_SEC WLAN_SEC_WPA2

Adafruit_CC3000 cc3000 = Adafruit_CC3000(ADAFRUIT_CC3000_CS, ADAFRUIT_CC3000_IRQ,
ADAFRUIT_CC3000_VBAT, SPI_CLOCK_DIVIDER);

Adafruit_CC3000_Client client;

void setup() {
  Serial.begin(9600);

  // For debugging, wait until the serial console is connected.
  delay(4000);
  while(!Serial);

  status_t wifiStatus = STATUS_DISCONNECTED;
  while (wifiStatus != STATUS_CONNECTED) {
    Serial.print("WiFi:");
    if (cc3000.begin()) {
      if (cc3000.connectToAP(WIFI_SSID, WPA_PASSWORD, WLAN_SEC_WPA2)) {
        wifiStatus = cc3000.getStatus();
      }
    }
    if (wifiStatus == STATUS_CONNECTED) {
      Serial.println("OK");
    } else {
      Serial.println("FAIL");
    }
    delay(5000);
  }

  cc3000.checkDHCP();
  delay(1000);
  Serial.println("Setup done");

  pinMode(8,INPUT);
}

void loop() {

  Serial.println("Test d'envoiement d'un SMS");
  delay(1000);

  TembooChoreo SendSMSChoreo(client);

```



```

// Invoke the Temboo client
SendSMSChoreo.begin();

// Set Temboo account credentials
SendSMSChoreo.setAccountName(TEMBOO_ACCOUNT);
SendSMSChoreo.setAppKeyName(TEMBOO_APP_KEY_NAME);
SendSMSChoreo.setAppKey(TEMBOO_APP_KEY);

// Set profile to use for execution

// Identify the Choreo to run
SendSMSChoreo.setChoreo("/Library/Twilio/SMSMessages/SendSMS");

SendSMSChoreo.addInput("AuthToken", "fd579eade81376e005372130696cf268");
SendSMSChoreo.addInput("From", "34960160592");
SendSMSChoreo.addInput("To", "34675394770");
SendSMSChoreo.addInput("Body", "ALARMA");
SendSMSChoreo.addInput("AccountSID", "AC3aacb2558b5c1b6232e2f5b95d7b250f");

// Run the Choreo; when results are available, print them to serial
SendSMSChoreo.run();

while(SendSMSChoreo.available()) {
  char c = SendSMSChoreo.read();
  Serial.print(c);
}
SendSMSChoreo.close();

Serial.println("\nEnviat\n");

}
Serial.println("Waiting...");
delay(60000); // wait 60 seconds between SendSMS calls
}

```

Codi 4. Sketch del sistema de seguretat del projecte sense l'utilització del sensor de moviment.

Aquest Sketch realitza les següents funcions descrites en 9 passos:

1. Declara totes les llibreries necessàries per a poder proveir-se de tots els recursos necessaris.
2. Declara les credencials Wifi i genera una instància que Choreo agafarà per a fer funcionar el sistema.
3. Realitza una serie de comprovacions per a validar que la placa WiFi s'ha connectat amb la xarxa indicada a les credencials
4. Inicialitza el choreo SendSMS
5. Agafa les credencials de Temboo i Twilio per a preparar-se per a enviar el SMS.
6. Envia el SMS
7. Retorna informació de l'enviament del SMS per a verificar aquest procés.
8. Tanca la connexió del Choreo.
9. Espera 60 segons per a tornar a enviar un SMS

En resum, aquest sketch ens envia un SMS cada 60 segons a un numero escollit.

4.2.3 Compilació del codi i execució del test (WiFi CC3000 + Tembo)

Un cop tenim generat el codi, es el moment de comprovar si el sketch que s'ha creat funciona com s'espera.

Pugem el codi a la placa, obrim la consola Serial d'Arduino, i verifiquem que tot funciona com s'espera. Efectivament obtenim una resposta positiva mitjançant la finestra serial d'Arduino IDE (fig. 37).

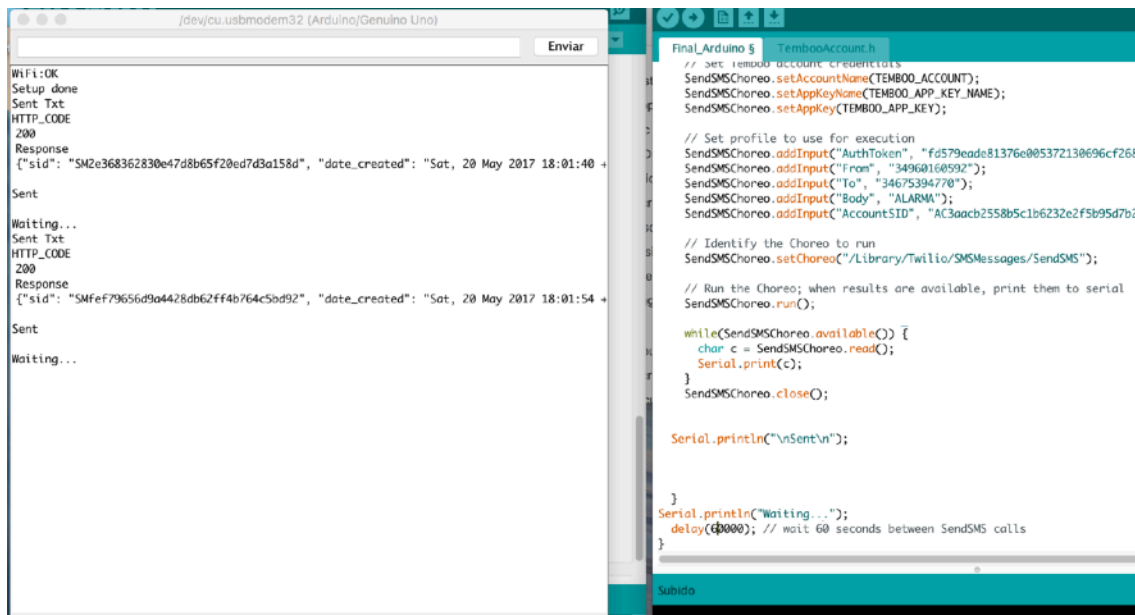


Figura 37. Demostració de funcionament del codi del sistema implementat utilitzant el sketch SendSms adaptat al nostre hardware

Es rep un SMS al mòbil destinatari (fig. 38).

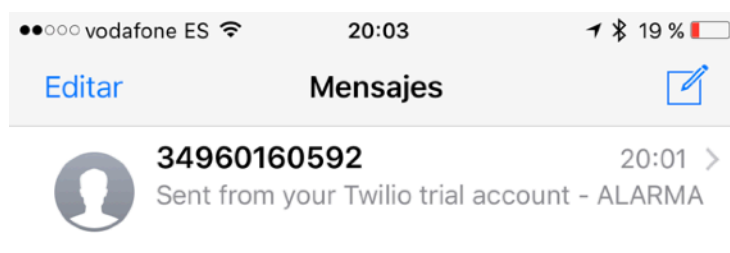


Figura 38. Demostració de recepció d'un SMS utilitzant el sketch generat

4.3 Implementació del sistema amb el sensor PIR

Un cop obtingut l'anterior sistema que envia un SMS a un numero escollit cada 30 minuts, quedarà afegir les funcions del sensor de moviment.

El següent sketch representa la solució completa del sistema de seguretat:

```
#include <SPI.h>
#include <Adafruit_CC3000.h>
#include <Adafruit_CC3000_Server.h>
#include <ccspi.h>
#include <Client.h>
#include <Temboo.h>
// Contains Temboo account information
#define TEMBOO_APP_KEY_NAME "SendSMSCristian"
#define TEMBOO_APP_KEY "oAUP058IFnmjGNC8*****" // Your Temboo app key
#define TEMBOO_ACCOUNT "hanero12"

#define ADAFRUIT_CC3000_IRQ 3
#define ADAFRUIT_CC3000_VBAT 5
#define ADAFRUIT_CC3000_CS 10

#define WIFI_SSID "Toby 2.4G" // cannot be longer than 32 characters!
#define WPA_PASSWORD "*****"
#define WLAN_SEC WLAN_SEC_WPA2

//variables definition sensor PIR
const int RedPin= 7;
const int GreenPin = 4;
const int SensorPin = 2;

Adafruit_CC3000 cc3000 = Adafruit_CC3000(ADAFRUIT_CC3000_CS, ADAFRUIT_CC3000_IRQ,
ADAFRUIT_CC3000_VBAT, SPI_CLOCK_DIVIDER);

Adafruit_CC3000_Client client;

int numRuns = 1; // Execution count, so this doesn't run forever
int maxRuns = 10; // Maximum number of times the Choreo should be executed

void setup() {
  Serial.begin(9600);
  pinMode (RedPin, OUTPUT);
  pinMode (GreenPin, OUTPUT);
  pinMode (SensorPin, INPUT);

  // For debugging, wait until the serial console is connected.
  delay(4000);
  while(!Serial);

  status_t wifiStatus = STATUS_DISCONNECTED;
  while (wifiStatus != STATUS_CONNECTED) {
    Serial.print("WiFi:");
    if (cc3000.begin()) {
      if (cc3000.connectToAP(WIFI_SSID, WPA_PASSWORD, WLAN_SEC_WPA2)) {
        wifiStatus = cc3000.getStatus();
      }
    }
  }
}
```

```

    if (wifiStatus == STATUS_CONNECTED) {
        Serial.println("OK");
    } else {
        Serial.println("FAIL");
    }
    delay(5000);
}
cc3000.checkDHCP();
delay(1000);
Serial.println("Setup done");
SendSMSChoreo.addInput("AccountSID", "AC3aacb2558b5c1b6232e2f5b95d7b250f");
// Identify the Choreo to run
SendSMSChoreo.setChoreo("/Library/Twilio/SMSMessages/SendSMS");
// Run the Choreo; when results are available, print them to serial
SendSMSChoreo.run();
while(SendSMSChoreo.available()) {
    char c = SendSMSChoreo.read();
    Serial.print(c);
}
SendSMSChoreo.close();
Serial.println("\nSent\n");

delay(1800000); // wait 30 minutes between SendSMS calls

}
Serial.println("Waiting...");
delay (10000);
}
pinMode(8,INPUT);
}
void loop() {
    int sensorread = digitalRead(8);
    int detector = digitalRead(SensorPin);
    // if((sensorread==HIGH)&&(alreadyread==0)){
    //alreadyread = 1;
    //initialtimegarageopen = millis();
    // }
    if ((detector==HIGH)){ //&&(millis()-initialtimegarageopen>=5000)
        Serial.println("Sent Txt!");
        delay(1000);

        TembooChoreo SendSMSChoreo(client);

        // Invoke the Temboo client
        SendSMSChoreo.begin();
        // Set Temboo account credentials
        SendSMSChoreo.setAccountName(TEMBOO_ACCOUNT);
        SendSMSChoreo.setAppKeyName(TEMBOO_APP_KEY_NAME);
        SendSMSChoreo.setAppKey(TEMBOO_APP_KEY);
        // Set profile to use for execution
        SendSMSChoreo.addInput("AuthToken", "fd579eade81376e005372130*****");
        SendSMSChoreo.addInput("From", "3496016*****");
        SendSMSChoreo.addInput("To", "3467539*****");
        SendSMSChoreo.addInput("Body", "ALARMA");
    }
}

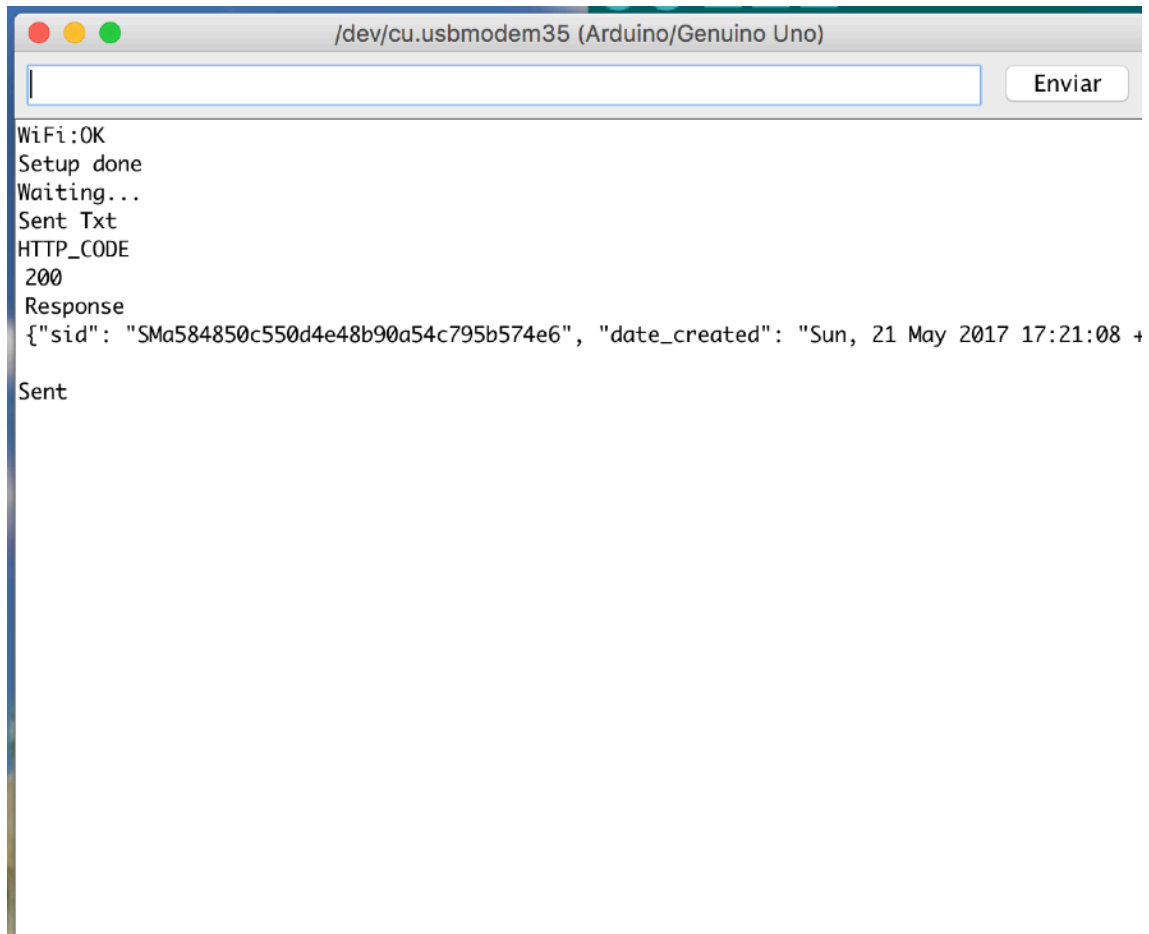
```

Codi 5. Sketch final del projecte. Sistema de seguretat.

4.4 Compilació del codi final i test

Aquest es l'apartat final en quant a implementació del codi. Es compila i carrega el codi generat dins la placa Arduino per a obtenir el sistema desitjat durant el projecte: un sistema de seguretat basat en Arduino.

El text que surt des de la consola Serial del programa Arduino IDE(fig. 39) mostra com el procés funciona correctament.



```
WiFi:OK
Setup done
Waiting...
Sent Txt
HTTP_CODE
200
Response
{"sid": "SMa584850c550d4e48b90a54c795b574e6", "date_created": "Sun, 21 May 2017 17:21:08 +
Sent
```

Figura 39. Demostració final del sistema implementat per al projecte.

A continuació, s'explica el procés que realitza el nostre codi:

- Primerament s'inicialitza la connexió WiFi i es mostra com es té accés a internet.
- El missatge "waiting..." indica que està esperant a que el sensor de moviment s'activi per a enviar el SMS.
- Després una persona passa per davant del sensor de moviment i el procés d'enviament de missatge s'activa tal com mostra el missatge "Sent Txt".
- Finalment ens retorna un codi 200 del servidor Twilio indicant que el missatge s'ha enviat correctament i ens retorna les credencials del missatge indicant la data i hora del missatge.
- Finalment s'envia el missatge Sent indicant que el procés ha finalitzar i que fins després de 30 minuts no es tornarà a activar el sistema.

S'adjunta un vídeo als documents del projecte demostrant el funcionament d'aquest (adj. demostració_1 i demostració_2).

I amb aquesta demostració queda finalitzada l'implementació de codi i de hardware.

4.5 Disseny encapsulament del producte

Finalment quedarà dissenyar un mètode d'emmagatzematge del nostre prototip de hardware per a crear un producte completament funcional, compacte i útil de cara a l'usuari. Es a dir, s'ha de trobar una capsula en la que es puguin emmagatzemar els diversos components del nostre producte:

- Placa Arduino UNO
- Protoboard
- Sensor PIR
- Espai per a passar cables.

Es necessitarà una capsula lleugera i resistent per a poder situar-la en una paret o sostre.

La capsula haurà de protegir correctament les connexions del sistema i els components en sí que formen part d'ell.

La capsula haurà de tenir un forat per a poder connectar-li el cable de corrent i el USB, ja que per canviar els valors de la xarxa WiFi on es connecta o les credencials de Twilio i Temboo, s'ha de modificar el Sketch manualment.

S'ha pensat en reutilitzar un mòdem vell en desús ja que reuneix les condicions necessàries descrites anteriorment, i addicionalment es fomenta el reciclatge de productes tecnològics guardats sense utilitzar.

En el nostre cas tenim una capsula d'un router "Vodafone" en desús(fig.40 a) que fa anys que es va canviar per un altre d'actual, i aquest va quedar guardat.

Obrim la capsa i buidem tot el que hi ha dins, deixant-ho buit(fig.40 b):



Figura 40. a) Fotografia d'un mòdem Vodafone antic en desús. b) fotografia d'un mòdem Vodafone antic obert i buit.

I introduïm els components dins i aprofitem les entrades que deixa la capsa per a introduir-hi els cables (fig. 41).

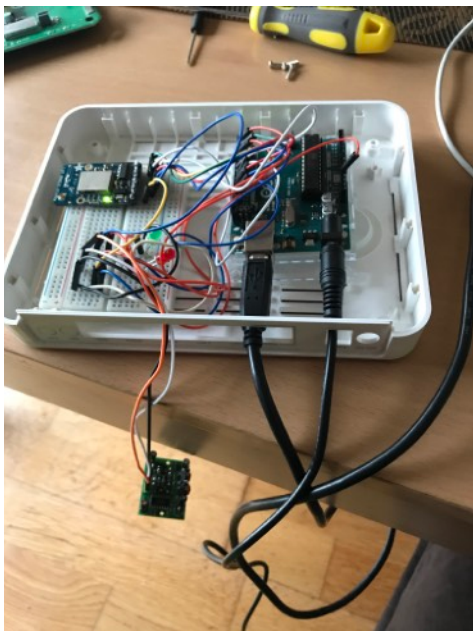


Figura 41. a) Mòdem Vodafone amb els components del sistema del projecte introduïts. b) Mòdem Vodafone amb els components del sistema del projecte introduïts mostrant el sensor PIR.

I el resultat és acceptable a nivell visual (fig. 42).

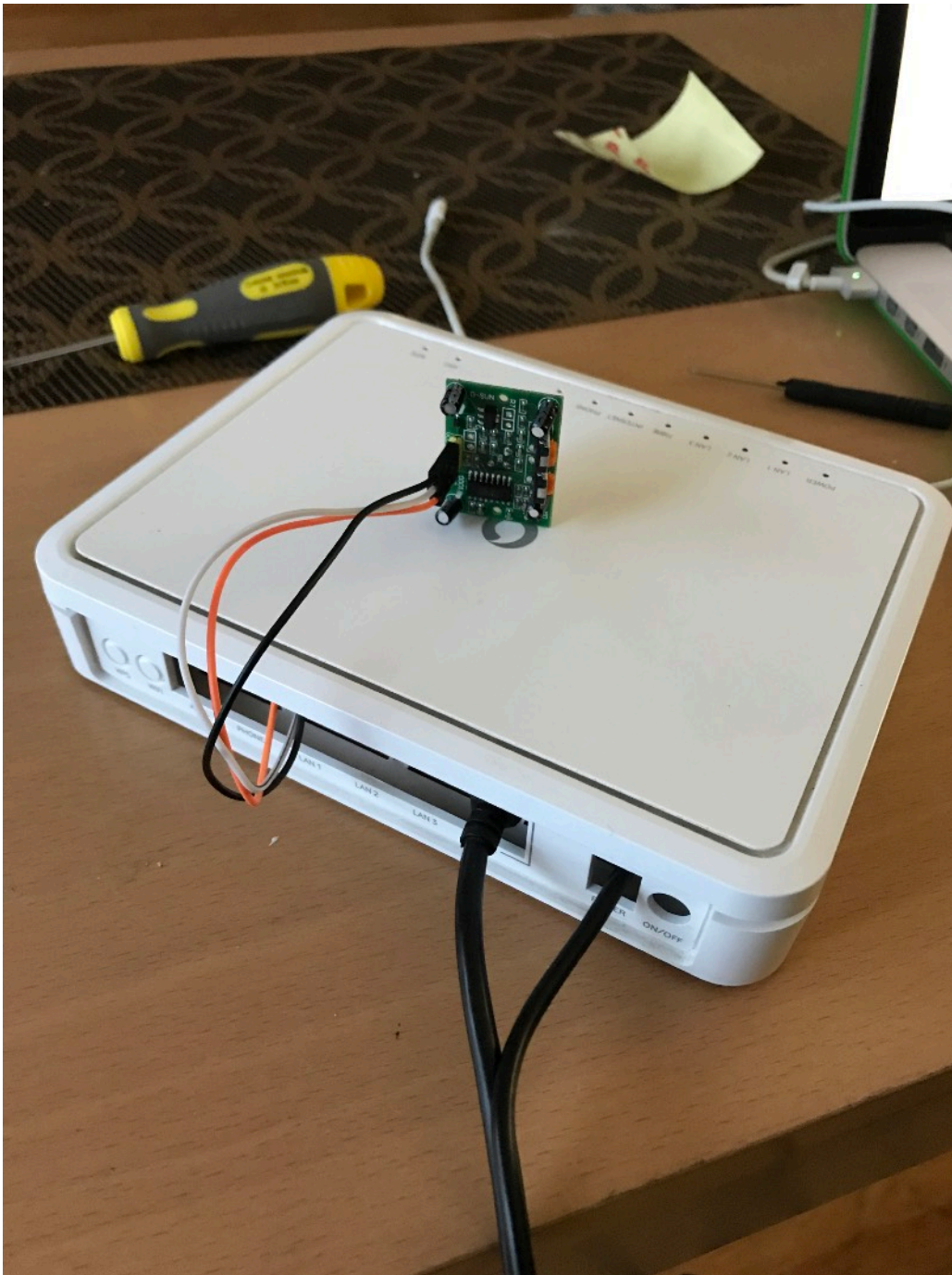


Figura 42. Sistema final implementat i encapsat sense el sensor PIR cobert

Falta realitzar el mateix procediment però amb el sensor PIR, que ha d'estar fora de la capsa per a detectar el moviment(fig. 542).

Per a recobrir el sensor PIR, aprofitem un envàs de 'petitsuisse', ho retallem fins que el component hi càpiga i l'introduïm dins. Llavors amb cinta aïllant ho recobrim tot, incloent-hi els cables, ho encaixem a la capsa gran, i el resultat és acceptable i fàcil de dur a terme(fig 43).



Figura 43. Producte final. Demostració completa. a) fotografia 1. b) fotografia 2.

5. Proves i resultats

5.1 Demostració

Un cop construït el producte final, queda demostrar la funcionalitat d'aquest mitjançant un procés de testing. Es crea un escenari real i es comprova durant un període de 24 hores que el sistema compleix amb la seva funció.

Es realitza l'instal·lació del producte com si d'un usuari final es tractés i l'utilitzés i es monitoritza la seva activitat amb l'objectiu de comprovar que compleix tots els requisits descrits al projecte.

S'adjunta un vídeo que demostra la funcionalitat del sistema mitjançant un escenari real. Es realitzen dos gravacions: "demostració_1" i "demostració_2".

Es comprova que el producte realitza correctament la seva funció, i ja estaria preparat per a ser distribuït.

5.2 Instruccions simplificades de muntatge del sistema

Finalment es conclou que un usuari es pot construir el seu propi sistema de seguretat seguint els següents passos:

3. Adquirir els següents components:
 - 3.1.Arduino UNO rev 3
 - 3.2.Adafruit CC3000
 - 3.3.Sensor PIR
 - 3.4.Cables de connexió i protoboard
 - 3.5.Una caixa on càpiguen tots els components un cop connectats
4. Descarregar el software Arduino IDE
5. Muntar el sistema seguint l'esquema de muntatge principal (fig. 37).
6. Donar-te d'alta a Temboo i obtenir les credencials de la compta
7. Donar-te d'alta a Twilio, obtenir els credencials de la compta i obtenir un número de telefon gratuït per a enviar SMS.
8. Afegir-li al sistema el Sketch adjuntat.
9. Modificar el valor de la xarxa WiFi (nom de la xarxa, contrasenya i tipus de seguretat del router) per els de la xarxa WiFi a la qual es connecti.
- 10.Modificar les credencials de la compta de Twilio per les teves.
- 11.Modificar les credencials de la compta de Temboo per les teves
- 12.Pujar el sketch Sistema_de_seguretat utilitzant l'aplicació Arduino IDE.
- 13.Connectar el sistema a una font d'alimentació.

S'ha calcul·lat que el temps de muntatge no hauria de superar les 3 hores, ja que solament s'ha de realitzar el muntatge físic segons l'esquema de muntatge(fig. 37) (1 hora) , donar-se d'alta a Temboo i Twilio (30 minuts), modificar el codi amb les teves dades i pujar-lo a la placa Arduino UNO(1 hora) i encapsar-lo (30 minuts).

La construcció doncs es viable per a un usuari amb un nivell mig de coneixements d'informàtica.

L'utilització del sistema es apte per a qualsevol usuari, independentment dels seus coneixements, degut a la seva senzillesa de funcionament (es rep un SMS a un numero desitjat quan s'activa el sensor de moviment).

5.3 Resultats i viabilitat del projecte

S'han realitzat proves de funcionament del sistema construït durant dos setmanes a fi d'analitzar l'acompliment dels requeriments inicials.

Monitoritzant l'activitat del sistema de seguretat s'han definit uns valors relacionats directament amb els requeriments. Aquests son els següents:

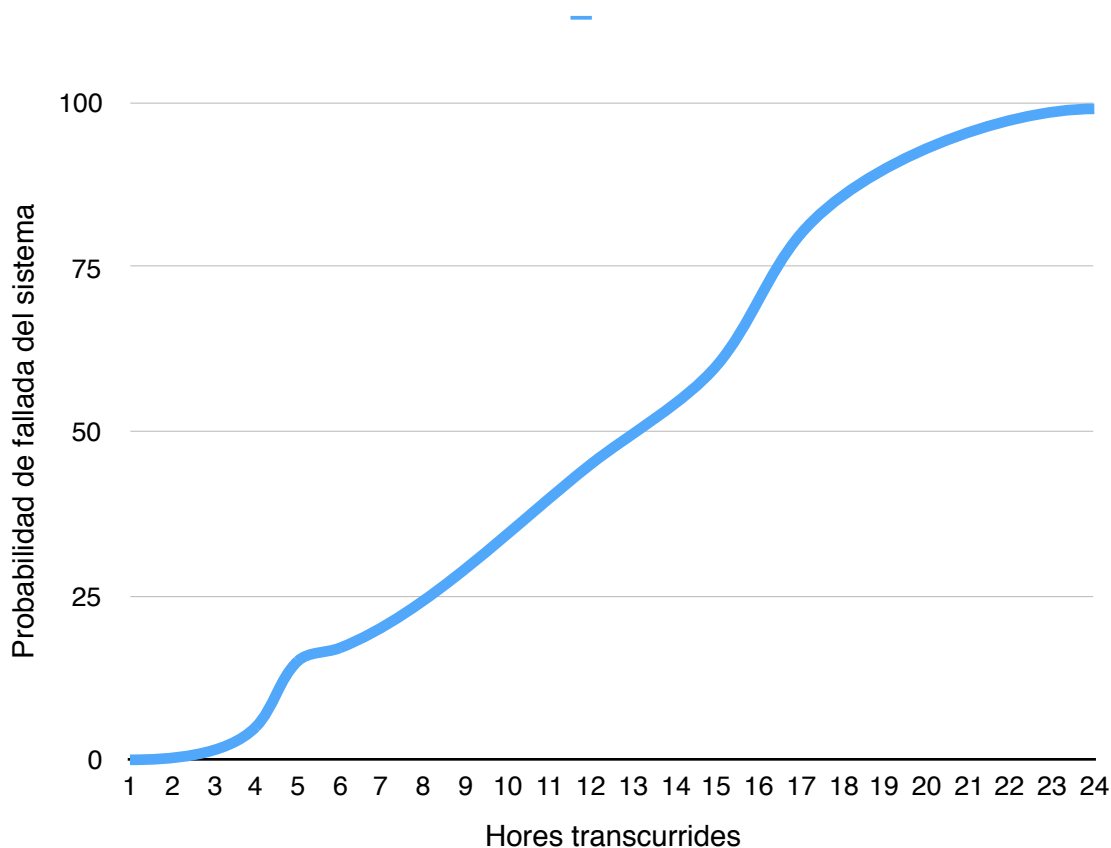
- Probabilitat de fallada electrònica
- Probabilitat de fallada de connectivitat
- Distància màxima recomanada
- Mitjana de temps consecutiu sense problemes de connectivitat

La següent taula relaciona la probabilitat de fallada del sistema amb el temps transcorregut durant la seva utilització:

Dia	Fallada electrònica	Fallada de connectivitat	Temps màxim consecutiu funcionant
1	No	no	24h
2	No	si	15h
3	No	si	8h
4	No	si	14h
5	No	no	24h
6	No	si	4h
7	No	si	12h
8	No	si	15h
9	No	si	18h
10	No	no	24h
11	No	si	10h
12	No	si	15h
13	No	si	6h
14	No	si	12h

Taula 1. Proves del producte

Amb el resultat de la taula anterior, s'ha elaborat un gràfic que relaciona la probabilitat de fallada segons la quantitat de temps funcionant.



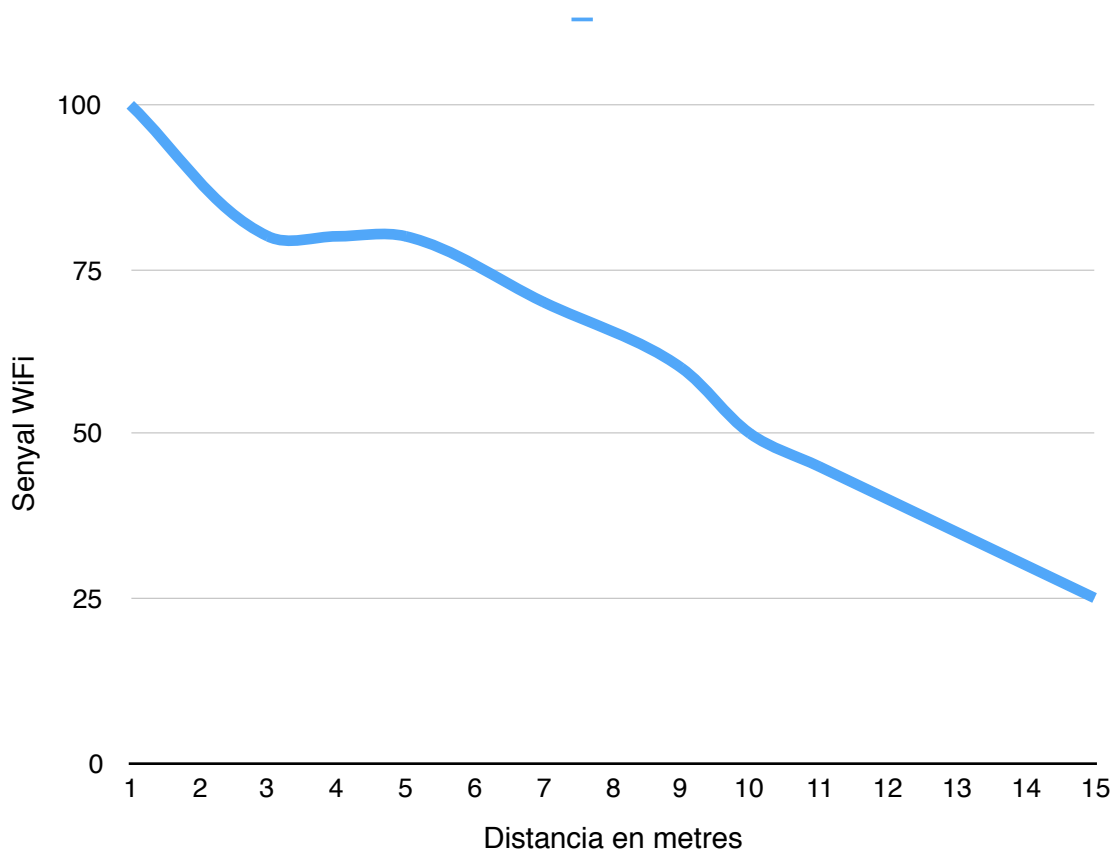
Gràfic 1. Probabilitat de fallada del sistema

Es pot apreciar que a partir de les 12 hores, la probabilitat de fallada comença a ser important, i el sistema deixa de ser fiable. Es conclou doncs, que el sistema pot funcionar de forma fiable durant una mitjana de 9 hores consecutives.

També es veu clarament com el sistema es altament fiable en quant a fallades electròniques. El sistema ha estat encès i funcionant en tot moment.

Durant aquest procés s'observa que si el punt d'accés perd la connexió a internet, el sistema deixa de funcionar fins que s'apagui i es torni a encendre. Aquesta es una de les causes principals de les fallades de connectivitat observades durant l'etapa de testeig del producte. L'altre causa es el simple fet de que el SMS enviat mai obté resposta i el sistema es queda indefinidament esperant al servidor Twilio.

S'ha realitzat un gràfic amb la distància màxima a la que pot estar el sistema respecte el punt d'accés.



Gràfic 2. Senyal de la placa segons distància

Per a ser més realista en el resultat, s'ha tingut en compte que per cada 4 metres hi ha una paret aproximadament.

Finalment es realitza un repàs dels requeriments inicials i es compara amb els obtinguts:

Requeriments inicials del projecte	Compliment de requeriments
Immediat: Es volia que el sensor envies un avís en un temps màxim de 2 minuts just després de detectar moviment.	El sistema envia el senyal en menys d'un minut
Fixe: El sensor de presència es situarà en un lloc fixe on monitoritzarà el moviment en tot moment. Encapsar el producte i fixar-lo no requerira de cap despesa econòmica extra, utilitzant materials reciclats per a acomplir aquest requeriment.	La capsula ha sigut construïda amb material reciclable. No ha suposat cap despesa econòmica extra
Connectat permanentment: el sistema pot funcionar permanentment, doncs tindrà un consum baix i estarà connectat a la xarxa WiFi. Es preveu que. La connectivitat a internet i WiFi del punt d'accés haurà de ser 100% fiable ja que sinó el sistema deixarà de funcionar i s'haurà de reiniciar manualment.	La connectivitat WiFi no es 100% fiable ja que si hi ha una fallada d'internet respecte el punt d'accés, el sistema deixa de funcionar fins que es reinicia.
Fiable: l'actuació del sistema es senzilla, detecta moviment, es connecta a internet i envia un SMS mitjançant algun servei web. És difícil un error en algun d'aquests punts descrits. Es preveu que el sistema mantindrà un percentatge de fallades d'un 10% degut al propi hardware i d'un 2% degut a problemes en quant a l'alimentació	El percentatge de fallades augmenta amb el temps d'ús, sent d'un 10% a les 5 hores de funcionament i d'un 25% passades les 9 hores. El percentatge de fallades degut a una fallada elèctrica del sistema s'aproxima a un 0%
Fàcil de fer servir: l'usuari final solament haurà d'encendre el sensor de moviment i aquest començarà a treballar permanentment. Si es vol desactivar, solament s'ha d'apagar la placa. La seva configuració consistirà en modificar les seves dades personals..	El sistema treballa de forma autònoma, i una vegada configurat no requereix de cap acció addicional per a fer-lo funcionar.
Fàcil d'instal·lar: El sistema solament s'ha d'instal·lar a una paret o sostre on el sensor de moviment apunti a l'àrea a detectar. També haurà d'estar connectat a la corrent mitjançant un cable. El temps de construcció i implementació no hauria de sobrepassar les 3 hores	S'ha elaborat un petit manual en el que s'estima que el temps de muntatge es d'aproximadament 3 hores. El sistema, mitjançant la capsula utilitzada, pot fixar-se a la paret fàcilment utilitzant claus.
Inalàmbric: la connexió del sistema a internet es totalment inalàmbrica, solament s'ha de connectar amb un cable a la corrent. el sistema haurà d'estar permanentment connectat sense problemes a internet de manera indefinida.	La connexió es realitza via WiFi. El sistema està connectat permanentment a la xarxa WiFi.

Taula 2. Compliment de requeriments inicials

Amb aquesta informació es demostra que el projecte és realitzable, ja que un usuari ho pot construir en unes 3 hores, el cost total del hardware es d'uns 60€ i el sistema és útil, tenint en compte que sol treballar durant unes 15 hores consecutives amb una probabilitat de fallada baixa.

Realitzant un petit anàlisi de mercat respecte a productes similars es pot comprovar ràpidament que no hi ha massa productes amb les mateixes característiques i que els que hi han, costen una mitja de 129€, més del doble que el nostre producte. La majoria de paquets d'alarma comercials solen costar uns 300€. Solament s'ha de realitzar una petita recerca a internet per a trobar aquestes solucions esmentades.

Amb el que es conclou, que el nostre producte, tenint en compte les limitacions que té, resulta ser viable tant a nivell econòmic com a nivell funcional.

6. Conclusions

La realització del projecte ha permès demostrar que un usuari pot construir el seu propi sistema de seguretat utilitzant la plataforma Arduino sense coneixements d'informàtica avançats. L'únic cost econòmic són els components físics, que, sumant el preu de tots els components, no superen els 60€. El codi utilitzat i els serveis oferts per Temboo i Twilio són open source. Amb el que es demostra que no es necessita contractar cap servei extern de pagament si es vol un resultat similar.

Fa aproximadament 10 anys era impensable que un usuari es pogués fabricar i programar els seus propis components com el descrit al projecte, ja que aquests productes estaven pensats per a producció en fàbriques programant les seves pròpies plaques. A més, el llenguatge de programació de les plaques era de més baix nivell, en canvi, si ens fixem ara, tenim llenguatges d'alt nivell com el que utilitza Arduino anomenat sketch.

Per a trobar la manera de crear el sistema final s'ha hagut de realitzar una feina d'investigació extensa, ja que no hi havia cap exemple de codi que mostrés una connexió entre el mòdul WiFi Adafruit CC3000 i les llibreries utilitzades per la plataforma Temboo utilitzant Arduino UNO. Per a dur a terme aquest objectiu s'ha hagut d'investigar el codi generat per Temboo de la llibreria SendSMS proporcionada per Twilio i les llibreries de la placa Adafruit CC3000 per a trobar algun punt de connexió entre els dos i fer funcionar el sistema dotant-lo de connectivitat per a realitzar l'enviament d'un SMS.

El projecte compleix amb tots els requeriments descrits al principi d'aquest document. Es pot concloure doncs, que s'han assolit els objectius del projecte de manera satisfactòria.

S'ha creat doncs, un sistema útil i fiable que li permetrà a l'usuari saber si algú ha entrat a casa seva de manera efectiva i simple. I aquest producte es pot construir seguint l'esquema de muntatge descrit en aquest projecte i copiant el codi adjuntat.

La lliçó més valuosa que s'ha apreciat i après durant la realització del projecte és, que una planificació adequada, es a dir, una correcta definició dels passos d'un projecte, és imprescindible per a garantir l'èxit d'aquest. Un millor anàlisi de requeriments al principi hages garantit una millor elecció en quant a la solució a implementar (primer es va escollir la solució de la càmera i després la del sensor de moviment).

Els diagrames de Gantt del projecte haurien d'haver sigut més detallats i precisos. D'aquesta manera potser s'hages detectat abans la inviabilitat del projecte durant el primer intent d'implementació, i s'hagués disposat de més temps en trobar una solució que compleixi els requeriments més optimitzada i sense tanta dificultat. De totes maneres la metodologia de la segona part del projecte, la reorientació, ha sigut un exit i l'ordre en que s'han realitzat totes les tasques ha permès que no es generí cap contratemps i que els passos seguits en la resolució hagi sigut lineal.

També s'ha après que, encara que un projecte resulti inviable, sempre hi ha una manera d'assolir l'objectiu principal buscant i executant alternatives equivalents.

El producte final ha sigut creat en el temps previst amb algunes variacions. Amb més temps per a la seva realització, es podrien haver implementat millores que haguessin fet el sistema final més fàcil de fer servir i útil.

A continuació, es proposa una llista de línies d'investigació que han quedat pendents d'implementar.

- Programar dins del sketch un sistema de control de connectivitat a internet que gestioni les possibles pèrdues de senyal amb la xarxa a que el sistema està permanentment connectat.
- Implementar un subsistema on se li afegeixi un altaveu connectat a la protoboard, el qual sol costar uns 3€ i es venen fàcilment per internet totalment compatibles amb Arduino UNO, i programar que realitzin un determinat soroll segons les vegades o freqüència amb la que s'activi el sensor de moviment.
- Crear un producte low cost, utilitzant productes de marques equivalents, com ara la pròpia placa Arduino UNO, o la placa WiFi. D'aquesta manera es podria construir un sistema de vigilància exactament igual al del projecte però amb un cost de menys de 20€.
- Implementar un sistema utilitzant una bateria que alimenti el nostre sistema. D'aquesta manera no faria falta cap mena de cable per a connectar el

nostre producte. S'hauria d'investigar, segons la duració de les bateries compatibles i consum de la placa, si seria viable aquesta implementació.

- Implementar un sistema a nivell de hardware per a encendre o apagar la placa Arduino UNO de manera còmode i fàcil.

Si no hagués existit cap contratemps durant la realització del projecte, probablement s'haguessin desenvolupat una o varies de les línies d'investigació descrites anteriorment. Encara que el resultat ha complert tots els objectius principals del projecte i ens trobem amb un producte final útil que compleix amb la seva tasca, tal com s'havia planificat.

Es demostra doncs, que amb Arduino, un usuari es pot construir el seu propi sistema de vigilància permanentment connectat amb pocs recursos i en poc temps.

Glossari

Arduino: placa de lliure configuració i programació amb la que es poden crear diversos projectes de diferent complexitat.

Arduino IDE: software compatible amb plaques Arduino que et permet crear codi i compilar-lo directament a la placa utilitzant el llenguatge sketch

Sketch: llenguatge de programació d'alt nivell utilitzat per a programar plaques Arduino

Adafruit CC3000: mòdul WiFi programable compatible amb arduino de l'empresa Adafruit.

Sensor PIR: component que actua com a sensor de moviment. Hi han models compatibles amb Arduino.

Mòdul: component acoblable a altres que realitza una funció concreta. En el nostre cas son components connectats a la placa Arduino

Temboo: empresa que proporciona diversos serveis, entre d'altres, la generació automàtica de codi Sketch compatible amb diverses plaques Arduino.

Chorus: codi generat per Temboo de manera automàtica que proporciona un servei escollit per l'usuari, com ara l'integració amb Twilio per a enviar SMS de forma automàtica, que és l'utilitzat en aquest projecte. El chorus utilitzat en el projecte s'anomena SendSMS, i es un sketch.

Twilio: empresa que proporciona serveis webs de telefonia, entre altres, l'enviament automàtic de SMS mitjançant un numero de telèfon que et proporcionen gratuïtament.

FIFO: de l'acrònim "First In Frist Out", es un tipus d'ordenació d'elements en el que el primer element que entra en un grup es el primer en sortir després. D'aquesta manera s'obté un tipus d'ordenació. Aquest algoritme es molt utilitzat en el mon de la programació, com per exemple en la reconstrucció d'imatges digitals a partir d'anal·lògiques..

Sistema de seguretat: protecció per les persones mitjançant elements que compleixen aquesta funció.

Fritzing: aplicació de codi obert de disseny per a la generació d'esquemes basats en Arduino.

Bibliografia

- [1] - Projecte existent de fabricació d'una càmera i Arduino UNO
<http://www.instructables.com/id/Arduino-Ethernet-Camera/>
8/03/2017
- [2] - Projecte existent de fabricació d'una càmera i Arduino Yun
<https://learn.adafruit.com/wireless-security-camera-arduino-yun/introduction>
8/03/2017
- [3] - Projecte existent de fabricació d'una càmera i arduino de manera inal·làmbrica
<https://openhomeautomation.net/wireless-camera/>
8/03/2017
- [4] - Esquema de connexions Arduino UNO
<https://www.arduino.cc/en/uploads/Main/arduino-uno-schematic.pdf>
22/03/2017
- [5] - Dataheet càmera OV7670
<http://www.voti.nl/docs/OV7670.pdf>
22/03/2017
- [6] - Tutorial configuració Arduino UNO
<https://www.arduino.cc/en/Guide/ArduinoUno>
2/04/2017
- [7] - Taula amb la definició dels connectors de la càmera OV7670
<http://www.arducam.com/camera-modules/0-3mp-ov7670/>
15/04/2017
- [8] - Esquema de connexió de la camera OV7670 amb Arduino UNO
<http://www.instructables.com/id/OV7670-Without-FIFO-Very-Simple-Framecapture-With-/>
15/04/2017
- [9] - Aplicació que connecta un ordinador amb Arduino UNO per a controlar la càmera OV7670
<https://cesarab.blob.core.windows.net/public/ReadSerialPortWin.exe>
24/04/2017
- [10] - Exemple de muntatge d'un sensor PIR amb Arduino
<http://www.prometec.net/sensor-pir/>
3/05/2017

[11] - Llibreria del component sensor PIR per a utilitzar en el programa Fritzing
<https://github.com/adafruit/Fritzing-Library/blob/master/parts/PIR%20sensor.fzpz>

3/05/2017

[12] - Documentació per el component Adafruit CC3000

<https://cdn-learn.adafruit.com/downloads/pdf/adafruit-cc3000-wifi.pdf>

8/05/2017

[13] - Pàgina de Temboo. Iniciació

<https://temboo.com/arduino/others/getting-started>

10/05/2017

[14] - Chorus de Temboo anomenat SendSms utilitzant el servidor Twilio

<https://temboo.com/arduino/others/send-sms>

11/05/2017

[15] - Pàgina de Twilio

<https://www.twilio.com/>

11/05/2017

[16] - Pàgina web Fritzing

fritzing.org

15/3/2017

[17] - Enllaç de les llibreries de codi de Adafruit per al component WiFi CC3000

<http://adafru.it/cHe>

17/05/2017

Annexos

I.1 Iniciació a Arduino UNO i Arduino IDE

Aquest annex explica com s'inicia la placa Arduino i com s'instal·la l'entorn.

Primerament connectarem la placa a la corrent i a l'ordinador:
Podem veure com el led prop de l'indicador "ON" es mostra de color verd, indicant que està encesa.

Després, es necessita la instal·lació d'un software de programació per a poder afegir-li codi a la placa i compilar-lo dins d'aquesta. La pàgina oficial d'Arduino et proveeix de software senzill i adaptat a totes les plaques d'aquest fabricant: Podem veure que tenim dues opcions(entorns) per a programar la placa (fig.8). En aquest cas s'ha escollit descarregar l'aplicació per a treballar en local. L'altre manera es executar el compilador via web, obtenint el mateix resultat, però depenent de connectivitat a internet.

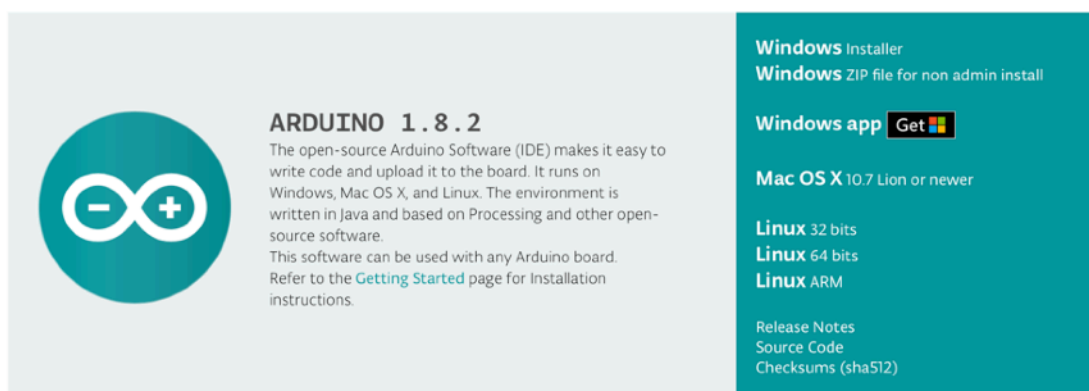
Access the Online IDE




ARDUINO WEB EDITOR
Start coding online with the [Arduino Web Editor](#), save your sketches in the cloud, and always have the most up-to-date version of the IDE, including all the contributed libraries and support for new Arduino boards. The Arduino Web Editor is one of the [Arduino Create platform's](#) tools.

[Try It Now](#)
[Getting Started](#)

Download the Arduino IDE



ARDUINO 1.8.2
The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.
This software can be used with any Arduino board. Refer to the [Getting Started](#) page for Installation instructions.

Windows Installer
Windows ZIP file for non admin install
Windows app 

Mac OS X 10.7 Lion or newer

Linux 32 bits
Linux 64 bits
Linux ARM

Release Notes
Source Code
Checksums (sha512)

Figura 8. Pàgina web de descarrega d'Arduino IDE [6]

El llenguatge de programació utilitzat es basa en sketch.

La programació de la placa Arduino s'estructura en dues parts(funcions) principals.

- “void setup()”: Aquesta funció s'executarà una sola vegada quan s'encengui la placa per primera vegada o quan es reiniciï.
- “void loop()”: Aquesta funció s'executarà en “bucle” constantment mentre la placa estigui encesa.

Abans de començar a introduir codi, hem de configurar el software Arduino IDE per a que es connecti correctament amb el nostre ordinador.

Des de l'aplicació Arduino IDE seleccionem la placa corresponent (fig.10).

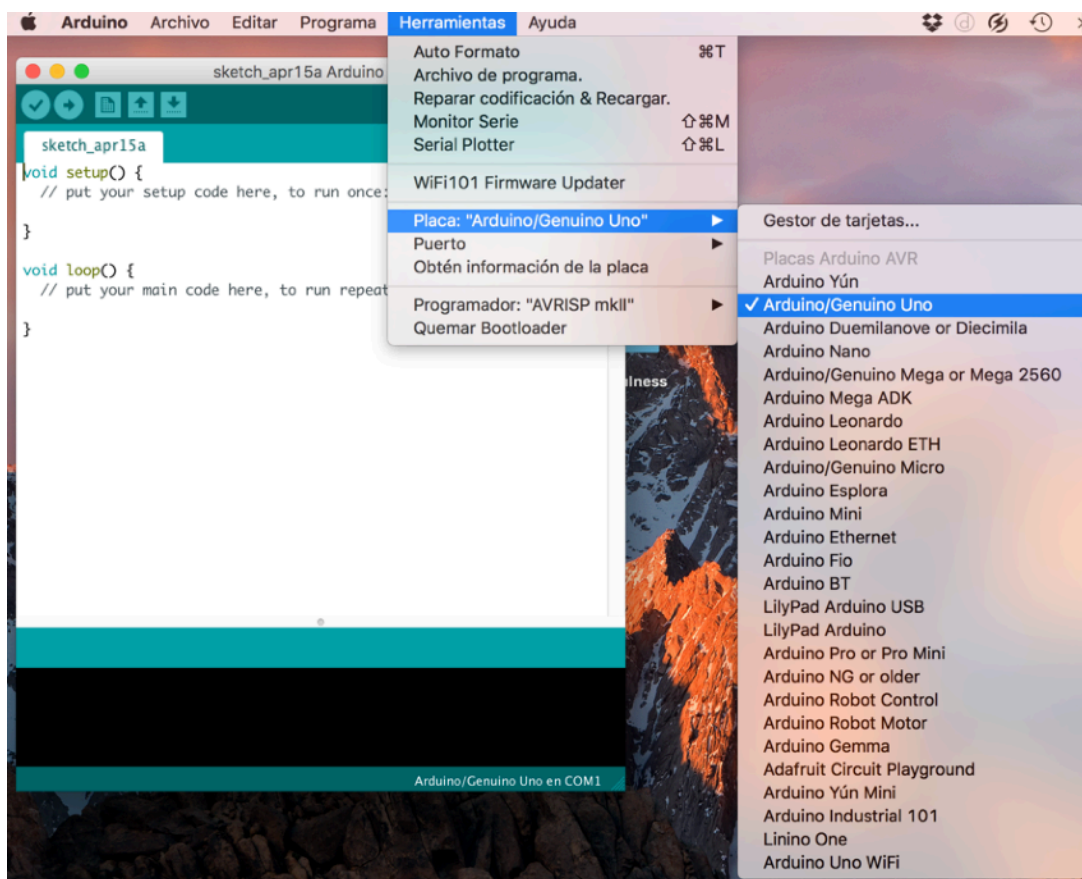


Figura 10. Selecció de placa en el programa Arduino IDE

Escollim el port el qual estem connectant la placa des de el nostre ordinador (fig. 11).

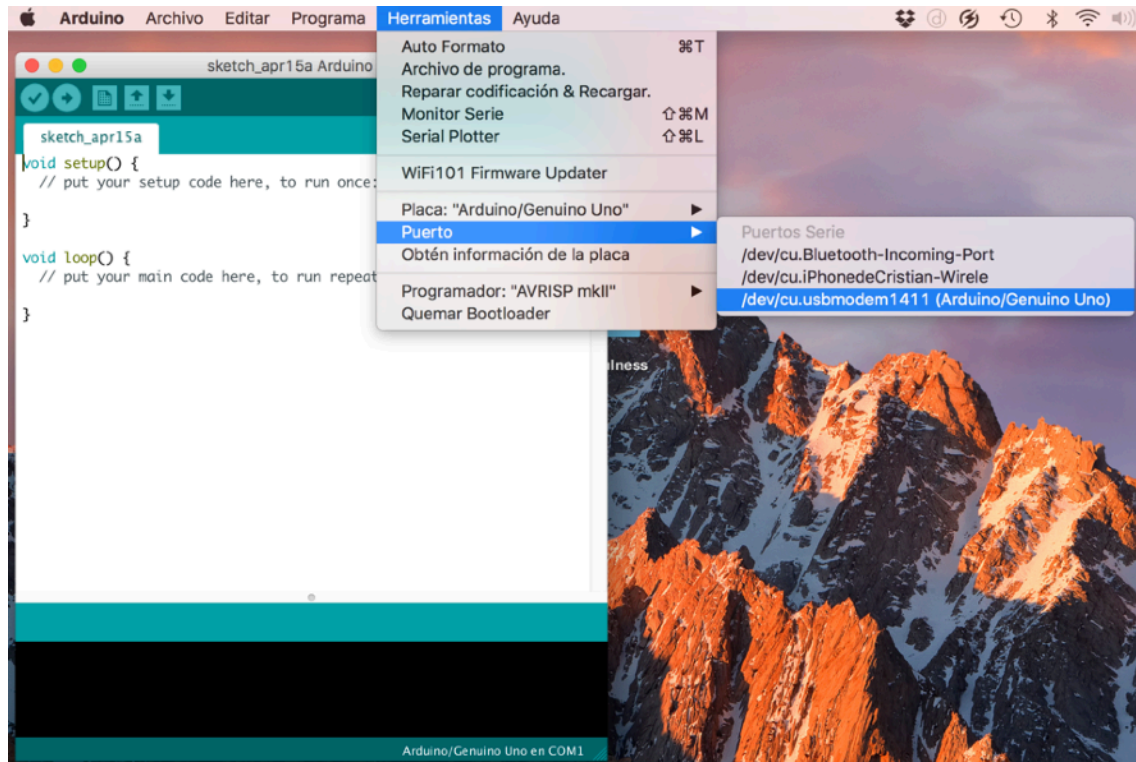


Figura 11. Selecció de port USB de sortida de Arduino IDE

Per a actualitzar la placa i que els canviïn es pugin, simplement s'ha de clicar al botó de pujar codi (fig. 12).

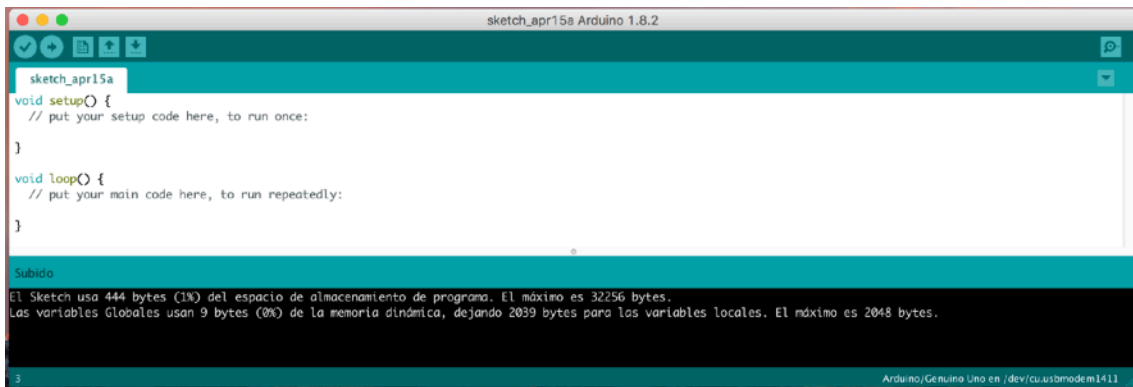


Figura 12. Demostració de compilació de codi en Arduino IDE

I.2 Inclusió de les llibreries del sistema a implementar

Aquestes llibreries(fig. 40) s'han d'incloure en el nostre fitxer de llibreries anteriors per a poder utilitzar-les. La carpeta en qüestió esta ubicada en: CarpetaDelprojecte/libraries/. Si el fitxer "libraries" no existeix, llavors s'ha de crear.

Si es vol utilitzar un altre directori on emmagatzemar les llibreries, s'ha d'anar a l'aplicació de compilació Arduino IDE, entrar en el menú Preferències i escollir la carpeta desitjada .

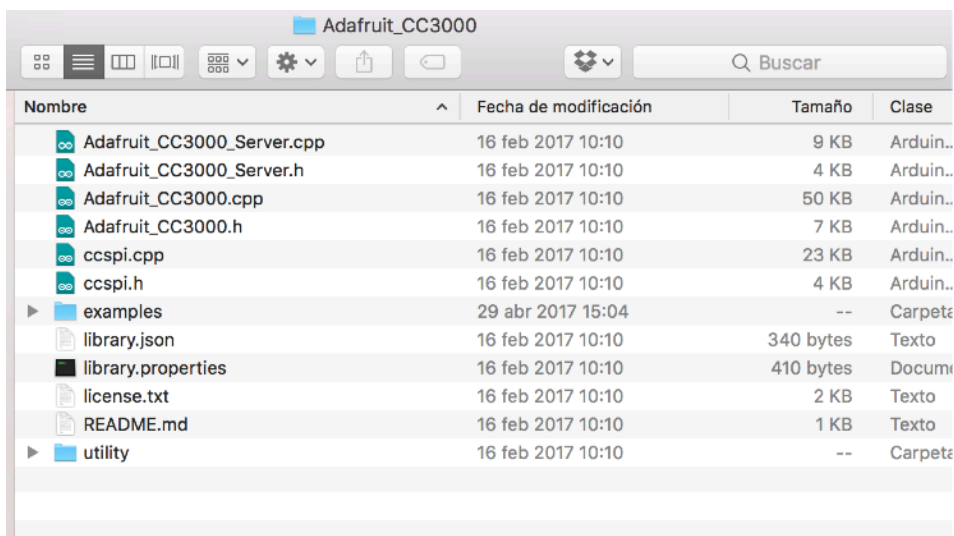


Figura 40. Captura de pantalla de les llibreries utilitzades per el component Adafruit CC3000

Hi ha un exemple de codi inclòs en aquesta llibreria anomenat "built code" que realitza un test complert de la placa WiFi amb Arduino. Es realitzarà aquest exemple per a comprovar el seu funcionament.

I.3 Anàlisi del codi sketch Adafruit CC3000

El codi queda més compacte i podem analitzar el comportament del modul Adafruit quan estableix connectivitat amb un router.

Si ens fixem en les funcions del “set up (void)” totes fan servir mètodes de l'instància anomenada “cc3000”, com per exemple:

```
if (!cc3000.begin())
{
  Serial.println(F("Unable to initialise the CC3000! Check your wiring?"));
  while(1);
}
```

Que comprova que la connexió amb la placa CC3000 s'estableix mitjançant la funció 'begin'.

O un altre exemple el poden extreure del següent fragment:

```
if (!cc3000.setDHCP()) {
  Serial.println(F("Failed to set DHCP!"));
  while(1);
}
```

Que comprova si s'ha creat una connexió DHCP.

Per tant podem deduir que a instància cc3000 es la que controla tota la connectivitat de la placa. I aquesta instància prové del següent fragment:

```
Adafruit_CC3000 cc3000 = Adafruit_CC3000(ADAFRUIT_CC3000_CS, ADAFRUIT_CC3000_IRQ,
Adafruit_CC3000_VBAT, SPI_CLOCK_DIVIDER);
```

Aquest fragment es clau en el funcionament de la placa CC3000.

Veiem que aquesta instància posseeix tota la informació necessària per establir connexió amb el router. De fet, se li passen variables les llibreries importades al principi.

I seguidament de la definició de les dades d'autenticació del punt d'accés podem establir aquesta connexió. Les següents línies defineixen aquesta informació:

```
#define WLAN_SSID      "SSIDXarxa Wifi"    // cannot be longer than 32 characters!
#define WLAN_PASS      "*****"
#define WLAN_SECURITY  WLAN_SEC_WPA2
```

Segurament aquestes dades son utilitzades per alguna de les llibreries que després utilitza l'instància “cc3000”.

Llavors podem concloure que l'instància anomenada "cc3000" es la que controla la connectivitat al complet de la placa Adafruit CC3000, i que juntament amb les llibreries incloses al codi anterior i la definició de les variables del router a connectar-se, es pot realitzar la transferència d'informació a través d'internet.

I.4 Anàlisi i deducció del codi compatible entre Temboo i el mòdul Adafruit CC3000

Llavors, el codi generat amb una placa Yun, no ens és vàlid, ja que no te cap funció que agafi cap credencial de la connectivitat del mòdul extern.

Però si utilitzem una placa diferent, com per exemple, Arduino Mega, i seleccionem el modul extern WiFi 101 Shield, llavors obtenim un codi que si ens pot ser útil, ja que conté traces de connectivitat manual cap a la xarxa via WiFi.

A continuació, es mostrarà el codi i eliminarem les traces que no estan directament relacionades amb la connectivitat de CC3000 a fi de trobar aquest enllaç.

```
#include <SPI.h>
#include <WiFi101.h>
#include <WiFiSSLClient.h>
#include <TembooSSL.h>
#include "TembooAccount.h" // Contains Temboo account information

WiFiSSLClient client;

void loop() {
  if (calls <= maxCalls) {
    Serial.println("Running SendSMS - Run #" + String(calls++));

    TembooChoreoSSL SendSMSChoreo(client);

    // Invoke the Temboo client
    SendSMSChoreo.begin();

    // Set Temboo account credentials
    SendSMSChoreo.setAccountName(TEMBOO_ACCOUNT);
    SendSMSChoreo.setAppKeyName(TEMBOO_APP_KEY_NAME);
    SendSMSChoreo.setAppKey(TEMBOO_APP_KEY);
    SendSMSChoreo.setDeviceType(TEMBOO_DEVICE_TYPE);

    // Identify the Choreo to run
    SendSMSChoreo.setChoreo("/Library/Twilio/SMSMessages/SendSMS");

    // Run the Choreo; when results are available, print them to serial
    SendSMSChoreo.run();

    while(SendSMSChoreo.available()) {
      char c = SendSMSChoreo.read();
      Serial.print(c);
    }
    SendSMSChoreo.close();
  }

  Serial.println("\nWaiting...\n");
  delay(30000); // wait 30 seconds between SendSMS calls
}
```

Si observem atentament, trobem una traça de codi exactament igual que la trobada anteriorment a les del Sketch del modul CC3000:

```
WiFiSSLClient client;
```

Aquesta traça està generant una instància, tal com es va generar amb el nostre component. Inclòs te el mateix nom.

```
TembooChoreoSSL SendSMSChoreo(client);
```

En aquesta línia veiem com la llibreria SendSMSChoreo ens està demanant el valor client per a poder treballar.

```
SendSMSChoreo.begin();
```

I de fet, amb aquesta funció, SendSMS comença a treballar.

Llavors si fem la següent comparació:

1- La traça de connectivitat del mòdul CC3000 es la següent:

```
Adafruit_CC3000 cc3000 = Adafruit_CC3000(ADAFRUIT_CC3000_CS, ADAFRUIT_CC3000_IRQ, ADAFRUIT_CC3000_VBAT, SPI_CLOCK_DIVIDER);
```

Es a dir, es la variable de connectivitat.

2- La traça de connectivitat d'aquesta llibreria de referència del mòdul Wifi101 es la següent:

```
WiFiSSLClient client;
```

Llavors, observant els mètodes de la llibreria del mòdul Adafruit, tenim una funció molt similar que estableix una instància de client:

```
Adafruit_CC3000_Client
```

La qual pot ser creada de la següent manera:

```
Adafruit_CC3000_Client client
```

I aquesta instància anomenada client, es justament la que necessita qualsevol Choreo de Temboo per a establir connexió. De fet, la utilitza d'aquesta manera:

```
TembooChoreoSSL SendSMSChoreo(client);
```

Però la nostra placa no utilitza una connexió SSL, tal com sembla indicar l'anterior funció, sinó una connexió simplement sense SSL.

Llavors analitzant la SDK de Temboo, podem trobar que hi ha un mètode molt similar, que ens demana aquesta variable, però que no utilitza SSL:

```
TembooChoreo SendSMSChoreo(client);
```

Que es la mateixa funció però sense les sigues "SSL".

I es probablement la funció que serà compatible amb la nostra placa Wifi CC3000 per a establir connectivitat a internet amb qualsevol llibreria Temboo, ja que la resta de codi es igual per a qualsevol altre component i mètode.

Per tant, unificant el codi de la nostra placa CC3000 i el codi de Temboo Choreo SendSMS el codi, en quant a la part d'establiment de connectivitat quedarà de la següent manera:

```

include <SPI.h>
#include <Adafruit_CC3000.h>
#include <Adafruit_CC3000_Server.h>
#include <ccspi.h>
#include <Client.h>
#include <Temboo.h>
#include "TembooAccount.h"

#define TEMBOO_APP_KEY_NAME "myFirstApp"
#define ADAFRUIT_CC3000_IRQ 3
#define ADAFRUIT_CC3000_VBAT 5
#define ADAFRUIT_CC3000_CS 10

#define WIFI_SSID "Toby 2.4G" // cannot be longer than 32 characters!
#define WPA_PASSWORD "Nuria1990"
#define WLAN_SEC WLAN_SEC_WPA2

Adafruit_CC3000 cc3k = Adafruit_CC3000(ADAFRUIT_CC3000_CS, ADAFRUIT_CC3000_IRQ,
ADAFRUIT_CC3000_VBAT, SPI_CLOCK_DIVIDER);

Adafruit_CC3000_Client client;

void setup() {

  Serial.begin(9600);

void loop() {

  Serial.println("funció d'enviament de SMS");
  delay(1000);
  TembooChoreo SendSMSChoreo(client);

```

I fins aquí parem de mostrar codi, simplement amb aquestes traçes, es mostra el mètode de connectivitat entre la placa CC3000 i una funció gestionada per Chorus.

La següent part de codi correspon a la funció de Send SMS, sense cap relació amb connectivitat amb qualsevol altre component. Es la part de codi que no s'ha de modificar. Encara que haurem d'adaptar el codi per a afegir-li les nostres credencials de Twilio per a que envii un SMS amb èxit.

Continuant amb l'anterior part del codi:

```

TembooChoreo SendSMSChoreo(client);
// Invoke the Temboo client
SendSMSChoreo.begin();

// Set Temboo account credentials
SendSMSChoreo.setAccountName(TEMBOO_ACCOUNT);
SendSMSChoreo.setAppKeyName(TEMBOO_APP_KEY_NAME);
SendSMSChoreo.setAppKey(TEMBOO_APP_KEY);
SendSMSChoreo.setDeviceType(TEMBOO_DEVICE_TYPE);

// Identify the Choreo to run
SendSMSChoreo.setChoreo("/Library/Twilio/SMSMessages/SendSMS");

// Run the Choreo; when results are available, print them to serial
SendSMSChoreo.run();

while(SendSMSChoreo.available()) {
  char c = SendSMSChoreo.read();
  Serial.print(c);
}
SendSMSChoreo.close();
}

Serial.println("\nWaiting...\n");
delay(30000); // wait 30 seconds between SendSMS calls
}

```

Analitzem les seves funcions:

Primerament, s'inicia la Choreo:

```

TembooChoreo SendSMSChoreo(client);
// Invoke the Temboo client
SendSMSChoreo.begin();

```

Un cop tenim completada la funció que agafa el component “client” , amb la següent , “begin” , es posa en marxa tot el procés del Choreo.

S'introdueixen les credencials de la nostra compta Temboo, les quals s'han de declarar en un fitxer anomenat “Account.h”, que després es declara com a llibreria a l'inici de l'aplicació. Per tant, el fitxer conté les següents traces:

```

#define TEMBOO_ACCOUNT "llanero12" // your Temboo account name
#define TEMBOO_APP_KEY_NAME "myFirstApp" // your Temboo app key name
#define TEMBOO_APP_KEY "FGINiAWzFJl8PmaQQc*****" // your Temboo app key

```

I a continuació el sketch principal les agafa:

```

// Set Temboo account credentials
SendSMSChoreo.setAccountName(TEMBOO_ACCOUNT);
SendSMSChoreo.setAppKeyName(TEMBOO_APP_KEY_NAME);
SendSMSChoreo.setAppKey(TEMBOO_APP_KEY);
SendSMSChoreo.setDeviceType(TEMBOO_DEVICE_TYPE);

```

El següent fragment agafa les nostres credencials de Twilio. I si el codi automàticament generat per Temboo, un cop introduïdes les nostres credencials, hagués funcionat, aquests valors serien els corresponents a la nostre compte de Twilio. Com no va poder ser, s'han d'emplenar manualment, quedant de la següent forma:

```
SendSMSChoreo.addInput("AuthToken", "fd579eade81376e00537213*****");
SendSMSChoreo.addInput("From", "349601****");
SendSMSChoreo.addInput("To", "34675*****");
SendSMSChoreo.addInput("Body", "ALARMA");
SendSMSChoreo.addInput("AccountSID", "AC3aacb2558b5c1b623*****");
```

D'aquesta forma, el mètode utilitzat per la Choreo SendSMS té les credencials necessàries per a poder enviar un SMS des de un telèfon origen a un de destí.

Solament ens queda l'execució del següent codi:

```
SendSMSChoreo.run();

while(SendSMSChoreo.available()) {
  char c = SendSMSChoreo.read();
  Serial.print(c);
}
SendSMSChoreo.close();

Serial.println("\nSent\n");
```

Aquest últim fragment correspon a l'enviament en sí del SMS, i a la lectura del "body" del SMS enviat, per a verificar que ha funcionat.

La última funció "close" tanca la connexió de tot el procés. Amb tota aquesta informació, s'ha creat un sketch de Choreo SendSMS compatible amb la nostra placa WiFi CC3000.

I.5 Passes per a la creació del codi SendSMS

Per a obtenir un sketch creat per Temboo, SendSMS, s'ha de seguir el següent procés:

- Donar-se d'alta a Twilio i obtenir un numero de telèfon, tal com s'ha explicat en l'apartat anterior.

- Dins de la pàgina de Twilio, s'han de guardar els camps "Account SID" i "AuthToken" que et proporciona la pròpia pàgina en la secció principal de l'usuari:

- En la pàgina de Temboo, a la choreo SendSMS, ens demana uns camps a emplenar, que són precisament els que hem copiat:

- AccountSID: codi copiat de la pagina Twilio
- AuthToken: contrasenya copiada de la pàgina Twilio
- Body: el contingut del SMS que s'enviarà (text).
- From: número de telèfon obtingut a Twilio en format internacional
- To: número de telèfon que rebrà el SMS, que és el de l'usuari que vol rebre l'avís de que l'alarma s'ha activat.

Un cop emplenats els camps es clica a "Generate Code" i en teoria obtenim el codi en format Sketch amb les dades d'autenticació del nostre usuari Twilio i els telèfons que enviaran i rebran el SMS. Però ens trobem amb un nou inconvenient (fig. 45).

Twilio . SMSMessages . **SendSMS** ☆

Sends an SMS to a specified phone number using the Twilio API.

INPUT Save Profile

AccountSID
The AccountSID provided when you signed up for a Twilio account.

AuthToken
The authorization token provided when you signed up for a Twilio account.

Body
The text of the message.

From
The purchased Twilio phone number, Twilio Sandbox number, or short code enabled for the type of message you wish to send (SMS or MMS). Format with a '+' and country code e.g., +16175551212.

To
The destination phone number. Format with a '+' and country code e.g., +16175551212.

▶ **OPTIONAL INPUT**

Generate Code ↻

▼ **OUTPUT** Error at 15:00 ET

A HTTP Error has occurred: The remote server responded with a status code of 404. Typically this indicates that the specified page could not be found on the server. The data returned from the remote server was: {"code": 20404, "message": "The requested resource /2010-04-01/Accounts/fd579eade81376e005372130696cf268/Messages.json was not found", "more_info": "https://www.twilio.com/docs/errors/20404", "status": 404}. The error occurred in the HTTPSend (Twilio - send SMS) step.

Figura 45. Example de generació de codi del Sketch SendSMS utilitzant el portal Temboo[14]

Per alguna raó desconeguda, el format del número de telefon no és compatible, i retorna un error. A conseqüència, no es genera codi sketch.

Per tant, per a continuar amb el projecte es procedeix a anal·litzar el codi de Choreo SendSMS sense adaptar la nostre placa WiFi ni les nostres dades de Twilio. Totes aquestes variables s'hauran d'implementar manualment al codi.

I.6 Codi Temboo Getting Started:

```
/*
  GetYahooWeatherReport

  Demonstrates making a request to the Yahoo Weather API using the Temboo Arduino Yun library.

  This example code is in the public domain.
*/

#include <Bridge.h>
#include <Temboo.h>
#include "TembooAccount.h" // contains Temboo account information

// the address for which a weather forecast will be retrieved
String ADDRESS_FOR_FORECAST = "104 Franklin St., New York NY 10013";

int numRuns = 1; // execution count, so that this doesn't run forever
int maxRuns = 10; // max number of times the Yahoo WeatherByAddress Choreo should be run

void setup() {
  Serial.begin(9600);

  // for debugging, wait until a serial console is connected
  delay(4000);
  while(!Serial);
  Bridge.begin();
}

void loop()
{
  // while we haven't reached the max number of runs...
  if (numRuns <= maxRuns) {

    // print status
    Serial.println("Running GetWeatherByAddress - Run #" + String(numRuns++) + "...");

    // create a TembooChoreo object to send a Choreo request to Temboo
    TembooChoreo GetWeatherByAddressChoreo;

    // invoke the Temboo client
    GetWeatherByAddressChoreo.begin();

    // add your temboo account info
    GetWeatherByAddressChoreo.setAccountName(TEMBOO_ACCOUNT);
    GetWeatherByAddressChoreo.setAppKeyName(TEMBOO_APP_KEY_NAME);
    GetWeatherByAddressChoreo.setAppKey(TEMBOO_APP_KEY);

    // set the name of the choreo we want to run
    GetWeatherByAddressChoreo.setChoreo("/Library/Yahoo/Weather/GetWeatherByAddress");

    // set choreo inputs; in this case, the address for which to retrieve weather data
    // the Temboo client provides standardized calls to 100+ cloud APIs
    GetWeatherByAddressChoreo.addInput("Address", ADDRESS_FOR_FORECAST);

    // add an output filter to extract the name of the city.
    GetWeatherByAddressChoreo.addOutputFilter("city", "/rss/channel/yweather:location/@city", "Response");

    // add an output filter to extract the current temperature
    GetWeatherByAddressChoreo.addOutputFilter("temperature", "/rss/channel/item/yweather:condition/@temp", "Response");

    // add an output filter to extract the date and time of the last report.
    GetWeatherByAddressChoreo.addOutputFilter("date", "/rss/channel/item/yweather:condition/@date", "Response");

    // run the choreo
    GetWeatherByAddressChoreo.run();

    // when the choreo results are available, print them to the serial monitor
    while(GetWeatherByAddressChoreo.available()) {

      char c = GetWeatherByAddressChoreo.read();
      Serial.print(c);
    }
    GetWeatherByAddressChoreo.close();

  }

  Serial.println("Waiting...");
  Serial.println("");
  delay(30000); // wait 30 seconds between GetWeatherByAddress calls
}
}
```

Codi 2. Exemple d'iniciació de Temboo

I.7 Datasheet Arduino UNO

A continuació es mostra el seu Datasheet (fig. 4):

Arduino™ UNO Reference Design

Reference Designs ARE PROVIDED "AS IS" AND "WITH ALL FAULTS". Arduino DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, REGARDING PRODUCTS, INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Arduino may make changes to specifications and product descriptions at any time, without notice. The Customer must not rely on the absence or existence of any features or instructions marked "optional" or "untested". Arduino reserves the right to change specifications and product descriptions at any time, without notice. The Customer must not rely on the absence or existence of any features or instructions marked "optional" or "untested". The product information on the Web Site or Materials is subject to change without notice. Do not reproduce a design with this information.

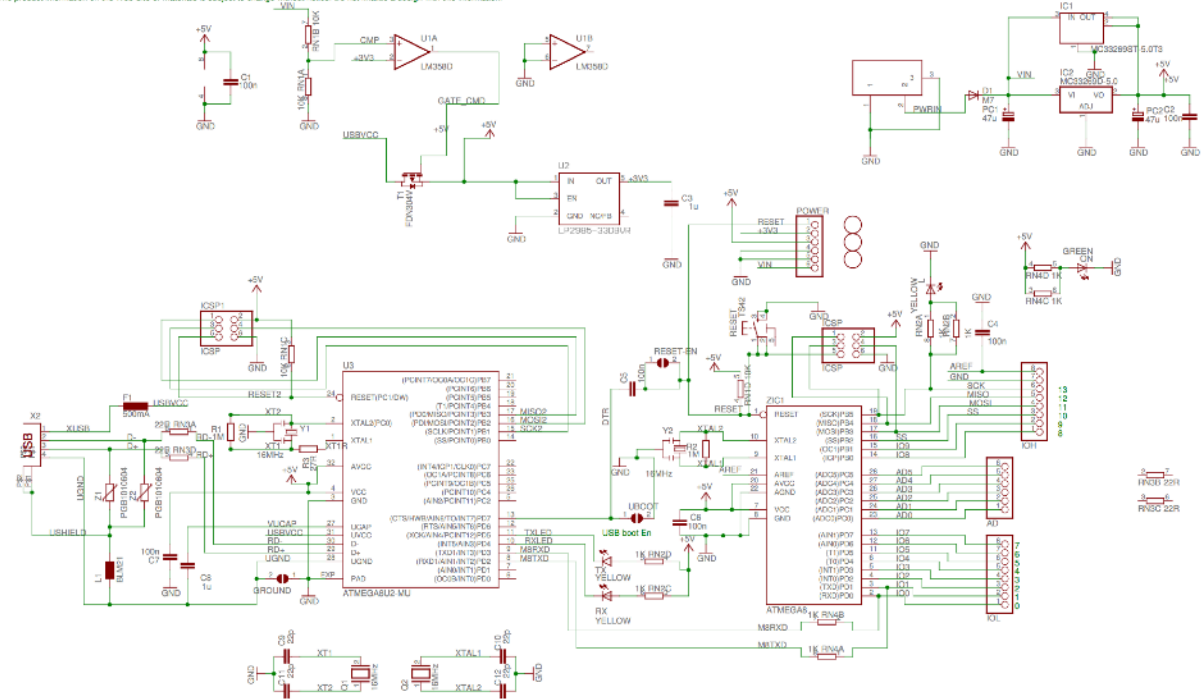


Figura 4. Datasheet Arduino UNO [4]

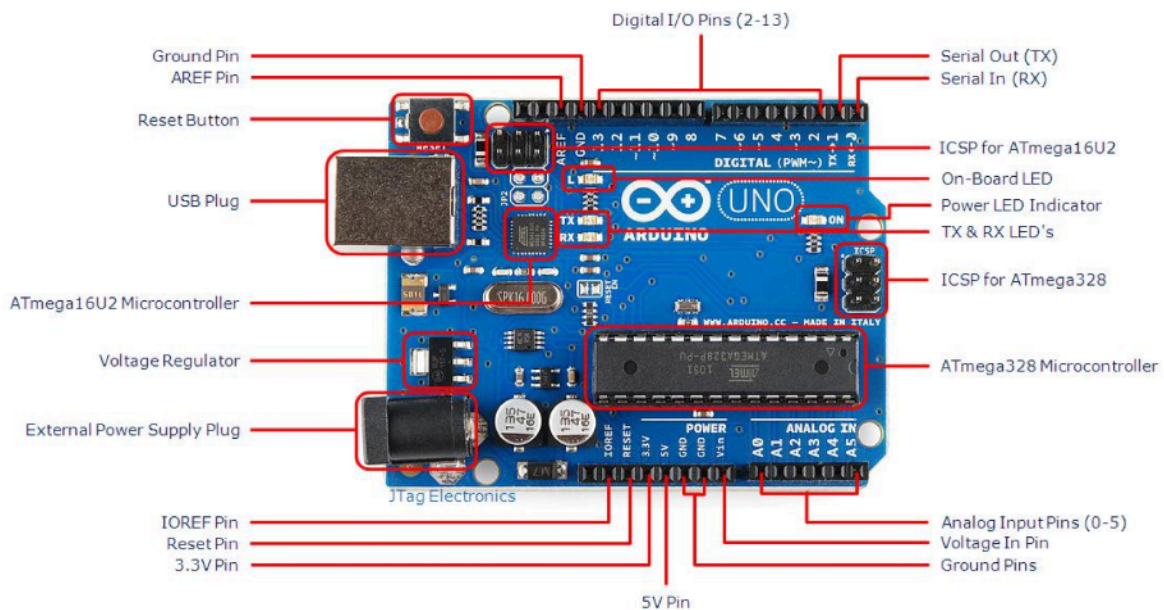


Figura 5. Esquema de connexions Arduino UNO [4]

I.8 Especificacions tècniques de la càmera OV7670

Es mostra les especificacions tècniques del component [5]:

Parameter:

- Photosensitive array: 640X480
- IO Voltage: 2.5V to 3.0V (internal LDO for nuclear power 1.8V)
- Power operation: 60mW/15fpsVGA YUV
- Sleep: <math><20\mu\text{A}</math>
- Temperature Operating: $-30\text{ }^{\circ}\text{C}$ to $70\text{ }^{\circ}\text{C}$
- Stable: $0\text{ }^{\circ}\text{C}$ to $50\text{ }^{\circ}\text{C}$
- Output Formats (8): YUV/YCbCr4: 2:2 RGB565/555/444 GRB4: 2:2 Raw RGB Data
- Optical size: 1/6 "
- FOV: 25°
- Maximum Zhen rate: 30fps VGA
- Sensitivity: 1.3V / (Lux-sec)
- SNR: 46 dB
- Dynamic range: 52 dB
- View Mode: Progressive
- Electronic Exposure: 1 line to 510 line
- Pixel Size: $3.6\mu\text{m} \times 3.6\mu\text{m}$
- Dark current: 12 mV / s at $60\text{ }^{\circ}\text{C}$

Features:

- High sensitivity suitable for illumination applications
- Low voltage suitable for embedded applications
- Standard SCCB interface compatible with I2C interface
- RawRGB, RGB (GRB4: 2:2, RGB565/555/444), YUV (4:2:2) and YCbCr (4:2:2) output format
- Supports VGA, CIF, and from a variety of sizes CIF to 40x30
- VarioPixel sub-sampling mode
- ISP has a compensation function to eliminate noise and dead pixels
- Support for image scaling
- Compensation for loss of optical lens
- 50/60Hz automatic detection
- Saturation automatically adjust (UV adjustment)
- Automatically adjust edge enhancement
- Automatically adjust the noise reduction
- Automatically affect the control functions include: automatic exposure control, automatic gain control, automatic white balance, automatic elimination of light stripes, automatic black level calibration image quality control including color saturation, hue, gamma, sharpness

L'esquema de la càmera indica la seva connectivitat [5]:

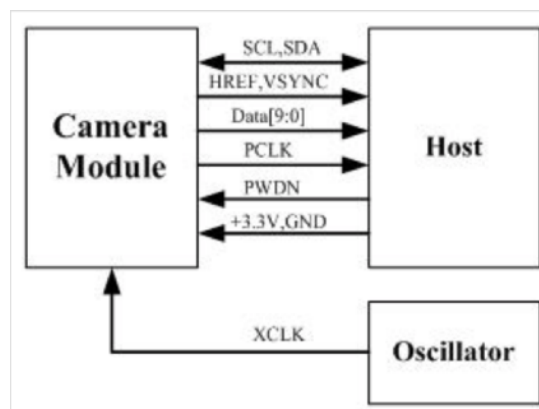


Figura 7. Datasheet del component OV7670

El següent esquema defineix la funció de cada connector del mòdul OV7670 [7]:

Pin No.	PIN NAME	TYPE	DESCRIPTION
1	VCC	POWER	3.3v Power supply
2	GND	Ground	Power ground
3	SCL	Input	Two-Wire Serial Interface Clock
4	SDATA	Bi-directional	Two-Wire Serial Interface Data I/O
5	VSYNC	Output	Active High: Frame Valid; indicates active frame
6	HREF	Output	Active High: Line/Data Valid; indicates active pixels
7	PCLK	Output	Pixel Clock output from sensor
8	XCLK	Input	Master Clock into Sensor
9	DOUT9	Output	Pixel Data Output 9 (MSB)
10	DOUT8	Output	Pixel Data Output 8
11	DOUT7	Output	Pixel Data Output 7
12	DOUT6	Output	Pixel Data Output 6
13	DOUT5	Output	Pixel Data Output 5
14	DOUT4	Output	Pixel Data Output 4
15	DOUT3	Output	Pixel Data Output 3
16	DOUT2	Output	Pixel Data Output 2 (LSB)

Figura 16. Definició dels pins del component OV7670 [7]

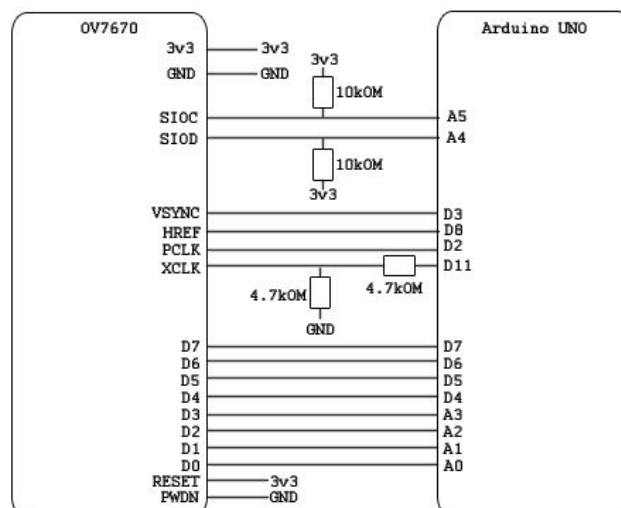


Figura 17. Esquema de muntatge de la càmera OV7670 amb Arduino UNO (Datasheet) [8]

I.9 Schematic Muntatge del sensor PIR(individual)

El seu Schematic (fig. 38) es genera automàticament gracies a l'aplicació fritzing. Només calen algunes modificacions respecte a la distribució dels connectors, ja que per defecte no es visualitzen de manera ordenada. El resultat obtingut es fàcilment analitzable.

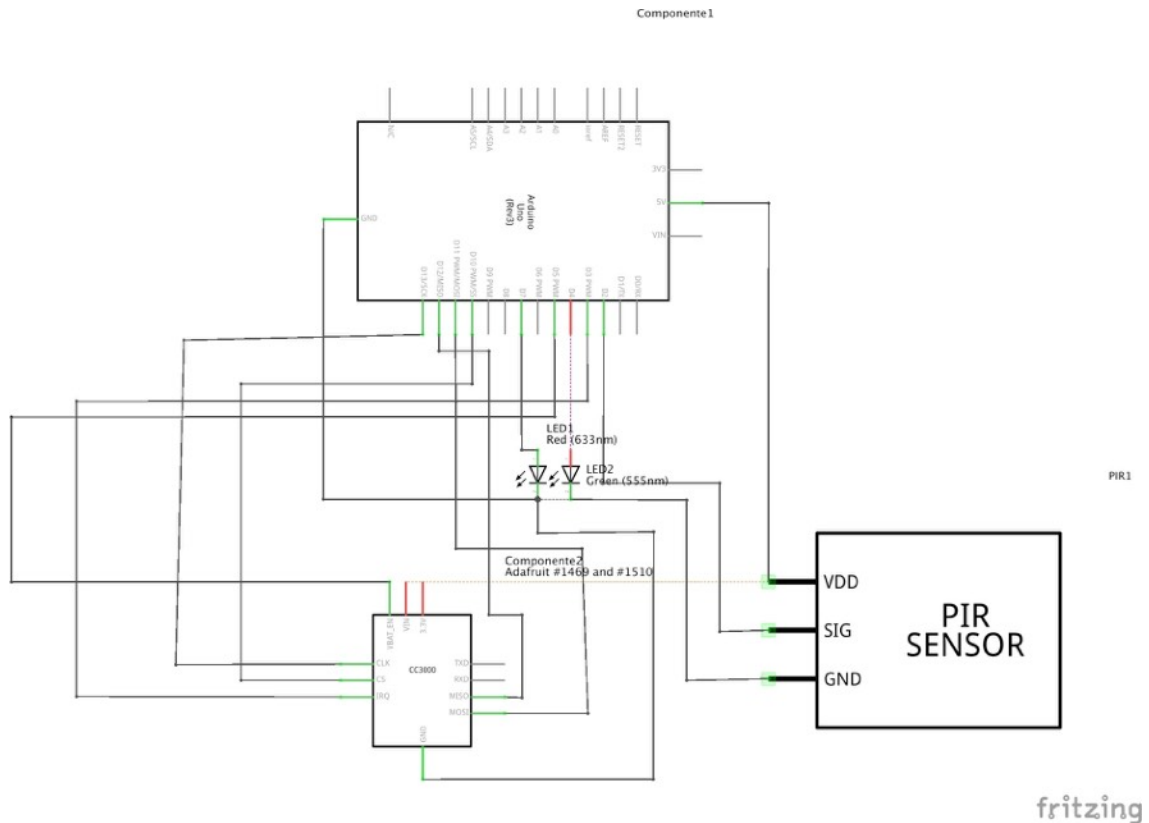


Figura 38. Datasheet realitzat utilitzant el sensor PIR, Arduino UNO i el mòdul Wifi Adafruit CC3000 per a crear el sistema final del projecte

I.100 Codi sketch de la càmera OV7670

```
/******  
This is a library for the openjumper TTL JPEG Camera (VC0706 chipset)  
  
These displays use Serial to communicate, 2 pins are required to interface  
*****/  
  
#include "camera_VC0706.h"  
  
void camera_VC0706::common_init(void) {  
    swSerial = NULL;  
    hwSerial = NULL;  
    frameptr = 0;  
    bufferLen = 0;  
    serialNum = 0;  
}  
  
// Constructor when using SoftwareSerial or NewSoftSerial  
#if ARDUINO >= 100  
camera_VC0706::camera_VC0706(SoftwareSerial *ser) {  
#else  
camera_VC0706::camera_VC0706(NewSoftSerial *ser) {  
#endif  
    common_init(); // Set everything to common state, then...  
    swSerial = ser; // ...override swSerial with value passed.  
}  
  
// 硬件串口构造函数  
camera_VC0706::camera_VC0706(HardwareSerial *ser) {  
    common_init(); // Set everything to common state, then...  
    hwSerial = ser; // ...override hwSerial with value passed.  
}  
  
// 初始化摄像头,通信波特率  
boolean camera_VC0706::begin(uint16_t baud) {  
    if(swSerial) swSerial->begin(baud);  
    else hwSerial->begin(baud);  
    return reset();  
}  
  
//复位  
boolean camera_VC0706::reset() {  
    uint8_t args[] = {0x0};  
  
    return runCommand(VC0706_RESET, args, 1, 5);  
}  
  
//动作检测  
boolean camera_VC0706::motionDetected() {  
    if (readResponse(4, 200) != 4) {  
        return false;  
    }  
    if (!verifyResponse(VC0706_COMM_MOTION_DETECTED))  
        return false;  
  
    return true;  
}  
  
//设置动作状态  
uint8_t camera_VC0706::setMotionStatus(uint8_t x, uint8_t d1, uint8_t d2) {  
    uint8_t args[] = {0x03, x, d1, d2};  
  
    return runCommand(VC0706_MOTION_CTRL, args, sizeof(args), 5);  
}
```

```

//获取动作状态
uint8_t camera_VC0706::getMotionStatus(uint8_t x) {
    uint8_t args[] = {0x01, x};

    return runCommand(VC0706_MOTION_STATUS, args, sizeof(args), 5);
}

//打开动作检测
boolean camera_VC0706::setMotionDetect(boolean flag) {
    if (! setMotionStatus(VC0706_MOTIONCONTROL,
        VC0706_UARTMOTION, VC0706_ACTIVATEMOTION))
        return false;
    uint8_t args[] = {0x01, flag};

    runCommand(VC0706_COMM_MOTION_CTRL, args, sizeof(args), 5);
}

//获取检测状态
boolean camera_VC0706::getMotionDetect(void) {
    uint8_t args[] = {0x0};

    if (! runCommand(VC0706_COMM_MOTION_STATUS, args, 1, 6))
        return false;

    return camerabuff[5];
}

//获取图片大小
uint8_t camera_VC0706::getImageSize() {
    uint8_t args[] = {0x4, 0x4, 0x1, 0x00, 0x19};
    if (! runCommand(VC0706_READ_DATA, args, sizeof(args), 6))
        return -1;

    return camerabuff[5];
}

//设置图片尺寸
boolean camera_VC0706::setImageSize(uint8_t x) {
    uint8_t args[] = {0x05, 0x04, 0x01, 0x00, 0x19, x};

    return runCommand(VC0706_WRITE_DATA, args, sizeof(args), 5);
}

/***** downsize image control */
uint8_t camera_VC0706::getDownsize(void) {
    uint8_t args[] = {0x0};
    if (! runCommand(VC0706_DOWNSIZE_STATUS, args, 1, 6))
        return -1;

    return camerabuff[5];
}
boolean camera_VC0706::setDownsize(uint8_t newsz) {
    uint8_t args[] = {0x01, newsz};

    return runCommand(VC0706_DOWNSIZE_CTRL, args, 2, 5);
}

/***** other high level commands */
//摄像头版本
char * camera_VC0706::getVersion(void) {
    uint8_t args[] = {0x01};

    sendCommand(VC0706_GEN_VERSION, args, 1);
    // get reply
    if (!readResponse(CAMERABUFFSIZ, 200))
        return 0;
    camerabuff[bufferLen] = 0; // end it!
    return (char *)camerabuff; // return it!
}

/***** high level photo comamnds */
void camera_VC0706::OSD(uint8_t x, uint8_t y, char *str) {
    if (strlen(str) > 14) {
        str[13] = 0;
    }
}

```



```

uint8_t args[17] = {strlen(str), strlen(str)-1, (y & 0xF) | ((x & 0x3) << 4)};

for (uint8_t i=0; i<strlen(str); i++) {
    char c = str[i];
    if ((c >= '0') && (c <= '9')) {
        str[i] -= '0';
    } else if ((c >= 'A') && (c <= 'Z')) {
        str[i] -= 'A';
        str[i] += 10;
    } else if ((c >= 'a') && (c <= 'z')) {
        str[i] -= 'a';
        str[i] += 36;
    }

    args[3+i] = str[i];
}

runCommand(VC0706_OSD_ADD_CHAR, args, strlen(str)+3, 5);
printBuff();
}

```

Contingut adicional

Sketch(codi) utilitzats en el projecte:

- Test_Arduino_UNO
- Camera OV7670
- Adafruit_CC3000
- Test_sensor_moviment
- Test_WiFi_Adafruit_CC3000
- Sistema_de_seguretat

Esquemes del projecte

- Arquitectura_del_projecte_1.ppx
- Arquitectura_del_projecte_2.ppx
- Esquema_primer_test_arduino_Uno.jpg
- Schematic_primer_test_arduino_Uno.jpg
- Esquema_sensor_PIR_prova.jpg
- Schematic_sensor_PIR_prova.jpg
- Esquema_sistema_final.jpg
- Schematic_sistema_final.jpg

Llibreries importades per als esquemes generats amb l'aplicació Fritzing

- Adafruit CC3000 WiFi Breakout.fzpz
- PIR sensor.pfpz

Videos demostratius

- demostracio_del_sistema1.mp4
- demostració_del_sistema1.mp4