# Projet de Fin d'Études:
# **Spectateur Augmenté**

*Teresa Tumbrägel*

IMA5

2015 – 2016

Tuteurs:

Thomas Vantroys

Alexandre Boé

# Table of Content

# Introduction

This report concerns a project carried out during the end of the winter semester 2015/2016 at Polytech Lille in cooperation with the research institution IRCICA. It represents the final year project of the department 'Informatique, Microélectronique et Automatique' (IMA) and will give an insight into the general context, the realization process and an outlook on the future.

The projects aim was to improve an already existing illuminated and interactive brassard and is a continuation of projects already carried out by former students of Polytech Lille and the IRCICA.

# Project Presentation

Usually the the interaction of artist and audience is just one way. This circumstance bears a great potential in regard on how to make live events more agile by opening up more then one way of interaction. The IRCICA developed an idea on how to change this and make it possible for the audience to be engaged in the actions of the artist. By doing that creativity can develop in a new dimension not accessible beforehand.

Therefore the audience is supposed to form a giant screen. Which is achieved by illuminating brassards with which every audience member is equipped. Each person then represents a single Pixel and all Pixels combined form an image. The devices are equipped with LEDs, micro controllers, an antenna and an accelerometer.

This project concerned itself primarily with the inclusion of the build in accelerometer, which can detect and interpret the movement of the carrier. By making the devices motion sensitive the direct reaction of the audience can be easily expressed. This project is an extension of an already existing project developed by Melanie Hautecoeur. Her project had the main objective to make radio communication between the brassards and fixed emitters possible in order to make the localization of single brassard possible. Furthermore an algorithm which processes the received data and can determine the position of the brassard was previously developed by the researchers of the IRCICA. Both research outcomes were accessible to me and used as references for my project.

# Material

All the appliances that were to be integrated are assembled on a board together with the micro-controller that makes it possible to access them.

For this project I used the following appliances with which the board is equipped:

- CC430F5137 micro-controller with, 32KB Flash and 4KB RAM
- MMA8653FCR1 accelerometer
- antenna
- two LED's

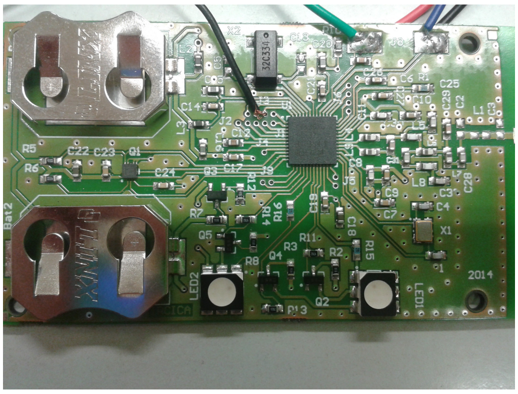


*Image 1: Board*

The board itself in turn was supplied by an MSP430, a common development board.

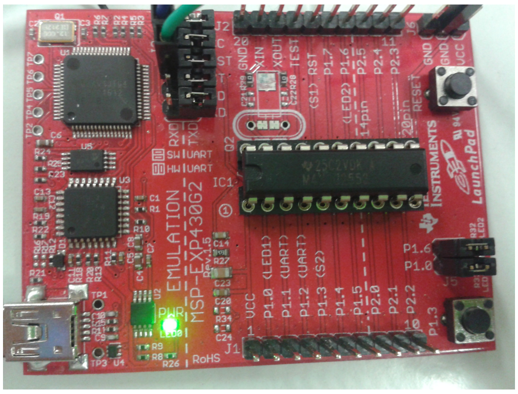


*Image 2: MSP430*

# Realization

The first step of the project was to get to know the material I was given and to understand the context and the previous work. I started out by programming codes that enabled me to control and manipulate the LED's of the board. By doing this I got a better understanding of the GCC environment and the MSP430 beforehand unknown to me.

Afterwards I concentrated on my main task which was to integrate further applications and to improve existing ones. Whereby one code built up on the previous ones. My next step, after the integration of the LED's, was to enable a serial communication between the micro-controller and the PC. This was followed by the development of an I²C communication which was necessary in order to integrate the build in accelerometer. The last point was the integration of communication between two boards. At first by a physical link and afterwards by radio.

The coding was done in C with the GCC compiler. For each executable program a Makefile, which determines compiling rules, and a main code, with the core program, is needed. These are accompanied by other pieces of code defining further self-defined or already existing programs. These other pieces of code always consist of a C-file, with the commands themselves, and a header-file, with the definition of the contained commands. The header-file has to be included in all other C-files that use the commands in order to tell the program how to use and where to find them.

The following is an example out of the *'led_commands.c'* file used to control the Led's.

```
void LED_1(void)
{
  set_color(0x10, 0x10, 0x10, 0x00, 0x00, 0x00);
}
```

In the header-file '*led_commands.h'* this piece is defined as *void set_color(uint8_t, uint8_t, uint8_t, uint8_t, uint8_t, uint8_t);*

If another code now wants to use this command it has to include *'led_commands.h'* in its header, the first lines of code.

The respective codes will be discussed more thoroughly below.

## LED

The commands defined in the LED's codes allow to manipulate the two build in LED's in regard to brightness and duration of illumination. A certain voltage should not be overstepped. Otherwise the LED's could be destroyed. This is done as follows.

*Timer_1=PWM*

A timer is assigned to each LED and to each timer a certain pulse-width modulation (PWM). The PWM defines for what part of a whole period a digital high or low is set for an exit. Since this happens very fast it results in a manipulation of the average output voltage of the exit.

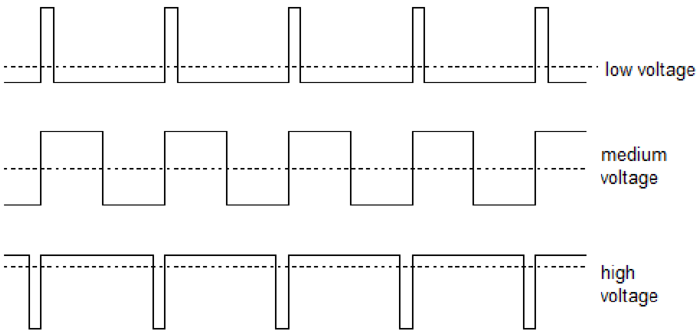



*Image 3: PWM*

Each LED has three different colors, red, green and blue, which can be used. For each color of each LED a further Timer is assigned.

*Timer_1.1= (red*PWM)/100*

*Timer_1.2= (green*PWM)/100*

*Timer_1.3= (blue*PWM)/100*

The variables *red*, *green* and *blue* define for how much of the whole time, now the PWM, the single colors of the LED's are illuminated. This has the same effect as the general PWM for the whole LED. It reduces the supplying voltage and therefore reduces the brightness. For example.

*Red=80*

*Timer_1.1= (80*PWM)/100=0.8*PWM*

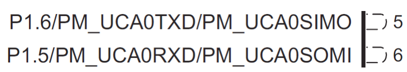
Now the red LED is illuminated for 80% of the whole time resulting in 20% less brightness.

The LED's can be turned of by setting the LED to 0% brightness. A blinking of the LED's was enabled by further function which turned the LED's on and off in certain time intervals.

## Serial Communication

The serial communication between the target and the PC enables the programmer to debug more easily. Therefore a USB interface and the Minicom, a text-based communication system for serial communication with Linux, is used. Data is transmitted to the PC over the USB interface and text elements are thereafter displayed on screen. The communication via USB is established by the use of Universal Asynchronous Receiver Transmitter (UART) communication. An UART is commonly used to connect a PC and a micro-controller

The micro-controller is equipped with transmit (TX) and receive (RX) pins. As you can see below.




A symbol is then send via the UART TX pin to the PC.

```
void uart_send(unsigned char byte){
while (!(UCA0IFG&UCTXIFG));
UCA0TXBUF = byte;
}
```

First the program ensures, that the signal paths is free, which means that nothing else is still being transmitted. Afterwards a signal is send.

```
void uart_send_string(char *string){
int i;
for(i=0;i<strlen(string);i++) uart_send(string[i]);
uart_send('\n');
uart_send('\r');}
```

When a whole string, a word, is supposed to be transmitted a loop calls the previous function several times. Each letter is still transferred one after another but it appears as one word on screen.

## *Accelerometer*

The brassard is also supposed to identify how engaged the carrier is in the event by motion evaluation. Meaning if he his for example clapping, standing up or dancing. Therefore it is supposed to determine when the carrier moves and how he moves. Evaluation of motion can be achieved with an accelerometer.

The MMA8653FCR1 is a commonly used capacitive accelerometer which detects changes in orientation. It can give orientation in reference to the three room directions here marked as x-, y- and z-axis.

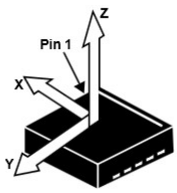



*Image 4: Orientation*

The information is thereafter given as a multiple of the earth gravity $g=9{,}81 m/s^2$.

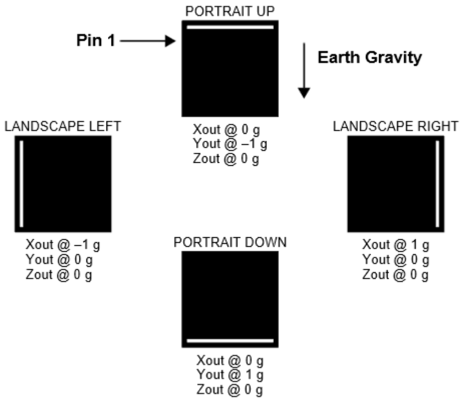


*Image 5: Sensitivity*

Furthermore it can give information about the acceleration of the device. This enables it to detect a free fall. When connected to a power supply it automatically wakes when the device is an free fall.

The accelerometer and the CC430 communicate by Inter-Integrated-Circuit (I²C). Therefore I²C communication had to be established first before the device could be integrated. I²C communication is carried out as illustrated in the following diagram. Where the first row illustrates the behavior of the receiving master and the second of the sending slave.

*Image 6: I²C communication*

1. ST-Device Address-W:

   The start message and the slave address have to be send by the master.

2. AK:

   Afterwards the slave acknowledges, that everything was transmitted.

3. Register Address:

   Next the register address, which is to be read is transmitted by the master.

4. AK

   The receiving also acknowledged by the slave.

5. SR-Device Address-R

   A repeated start in combination with the slave address is executed which indicates the slave that the data can be send.

6. AK-DATA

   The slave acknowledges again the correct reception and transmits the actual data.

7. AK

   In the end the communication terminates with an acknowledgment by the master which signals that everything was carried out correctly.

In this project the micro-controller was the master and the accelerometer the slave. The accelerometer is able to detect movement three dimensionally. For each dimension a specific register, here the 'reg_no' is reserved. In the main function the following I²C-function is called once in every loop iteration for each dimension.

```
        int I2Cgetbyte(unsigned char reg_no) {
1.      UCB0CTL1 |= UCTXSTT + UCTR;
2.      while(!(UCB0IFG & UCTXIFG));
3.      UCB0TXBUF = reg_no;
4.      while((UCB0CTL1 & UCTXSTT) && !(UCB0IFG & UCNACKIFG));
A.      if(UCB0IFG & UCNACKIFG){
        UCB0CTL1 |= UCTXSTP;
        while((UCB0CTL1 & UCTXSTP));
        return -1;
        }
        while(!(UCB0IFG & UCTXIFG));
        UCB0CTL1 &= ~UCTR;
5.      UCB0CTL1 |= UCTXSTT;
6.      while((UCB0CTL1 & UCTXSTT) && !(UCB0IFG & UCNACKIFG));
B.      if(UCB0IFG & UCNACKIFG){
        UCB0CTL1 |= UCTXSTP;
        while((UCB0CTL1 & UCTXSTP));
```

```
return -1;
 }
7.  UCB0CTL1 |= UCTXSTP;
    while (!(UCB0IFG & UCRXIFG));
    unsigned char data = UCB0RXBUF;
    while((UCB0CTL1 & UCTXSTP));
     return data;
    }
```

A. and B. represent exit conditions if an error occurs. At the end the from the slave transmitted data is returned to the function call. In our case the data is the specific orientation. The function is for example called as follows.

```
ax=I2Cgetbyte(0x1);
ay=I2Cgetbyte(0x3);
az=I2Cgetbyte(0x5);
```

The value transmitted via UART-communication and displayed on screen afterwards.

```
sprintf(output,"%02x %02x %02x",ax,ay,az);
uart_send_string(output);
```

Whereby the values transmitted by the accelerometer must first be converted from binary to hexadecimal numbers to make it more easily interpretable for the human user.

## Board to Board Communication

Again a UART communication was used for the first communication between two boards. Therefore a direct physical link was established by connecting the TX and RX pins of the two boards. Whereby the TX of one was connected with the RX of the other and vice versa.

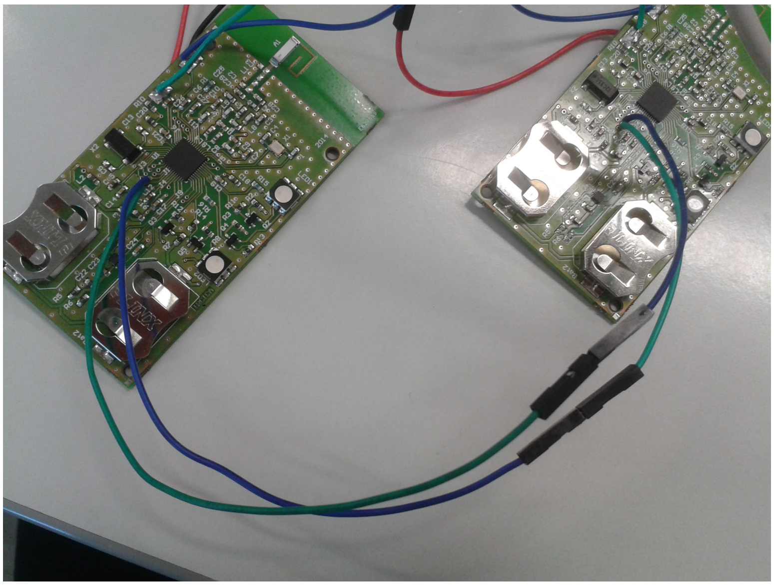


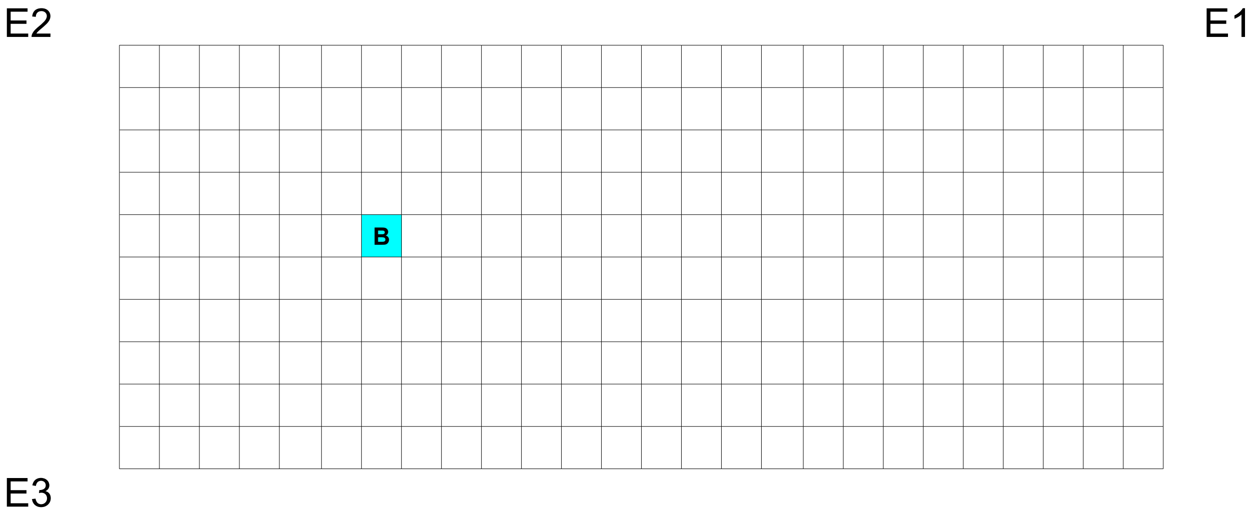
*Image 7: Serial Communication*

First a simple echo program, where one board echos back what it received from the other board, was established. This receiving and sending was accompanied by LED's which signaled the current program status of the respective board.

Afterwards the boards were supposed to communicate by radio. Each board is equipped with an antenna to enable this. With the radio communication a localization of the brassards is made possible.

A board can be used either as an emitter or as a receiving brassard. Whereby the emitter use is just one way communication.

Three emitters are placed in a room. They emit first one after another and then in alternating pairs. The brassards receive the signals. Depending on the position of the brassard in the room each signal is received with a different strength. The algorithm of the IRCICA researchers can then evaluate these signal strengths and determine how far away the brassard is located from each emitter and thereby be located. The setup is illustrated in the following schema. Every square represents a seat in the room and the emitters are placed around it. One example brassard is also illustrated.



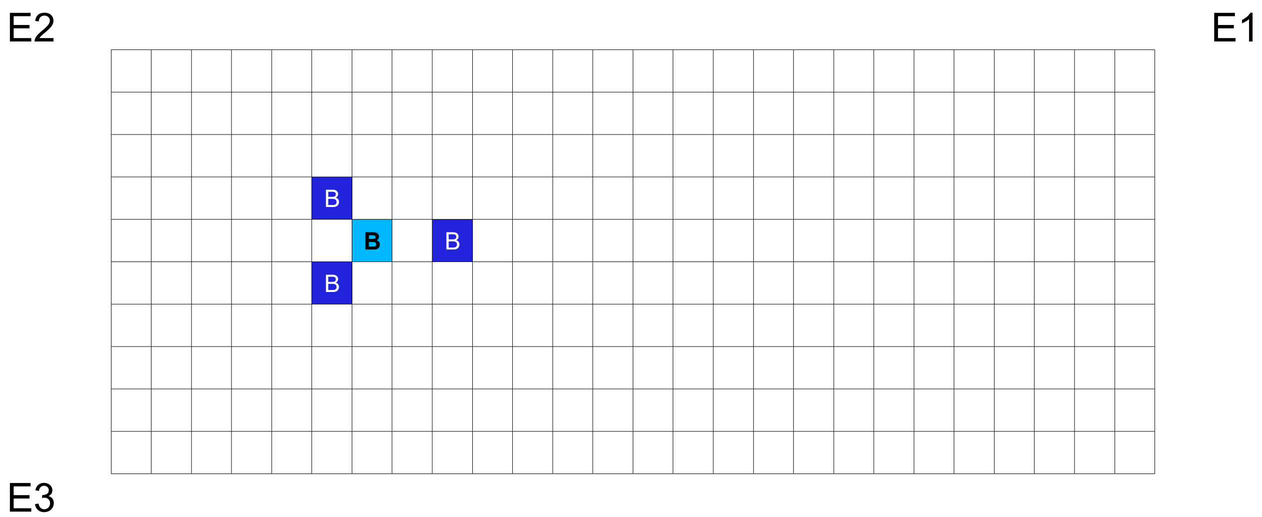

The emitters will send signals as follows.

- E1
- E2
- E3
- E1 and E2
- E2 and E3

Every room direction is thereby covered two dimensionally. With a fourth emitter a third room dimension can also easily be added.

In the final application this makes it possible to create an accurate image. When it is known where each brassard is located a concrete Pixel of the whole image can be assigned to it. It also ensures, that the image is not disturbed when a brassard, which means the person wearing it, moves.

Beacause of errors, which occur for example due to obstacles or interference, the localization is not completely accurate. The localization can be further improved by radio communication between the boards. A higher accuracy can be achieved when the boards detect one another. Therefore the next neighbors of the brasards are localized and the position relative to the neighbors is combined with the position determined by the signal strength of the emitters.




This part of the project was largely realized with the previously developed coding. One board hereby acted as emitter and the other as receiver. On a defined frequency the first board sends a data packed and the other receives this package. Whereby it is important that the transmitted data is actually read by the recipient. Otherwise the receiving channel will not be free for further reception.

# Following work

The work can be continued in several points.

- Implementation and later testing of the localization with the radio communication. This still has to be done for the absolute position in the room and the position relative to the neighbors.
- Accuracy improvement based on the localization.
- Tests with a higher number of boards.
- Finally the display of an image.

# Conclusion

During the last months of project work I gained an insight in the complexity of micro-controller programming which enabled me to improve my development abilities in several fields.

In the beginning I started out by familiarizing with the project. The first step was to get accustomed with the development environment. Even though I was already familiar with micro-controller programming the framework at hand was completely new to me. Beforehand I worked primarily with assembler when programming a micro-controller. Which proofed to be a great difference. Also the unknown programming style and unknown set of commands were something I first had to grow accustomed to. In effect this slowed down the overall advancement of the project.  Simultaneously I got to know the project work of my predecessors. I found that they had already made great advancements. Later on their findings helped me to advance parts of my own project. Some of them I was also able to integrate directly. Unfortunately it often proofed to be difficult to retrace the previous coding and therefore to integrate it right away. In effect the reconstruction of the code build up was also very time consuming. Overall the first project stage was the most labor-consuming part of the project but did not produce the most actual output. Nevertheless I think for me personally it was the part were I learned the most and made the most progress in my personal development. I improved in gathering necessary information and acquiring new knowledge myself.

The later project stages where somewhat more rewarding and I was able to implement several codes and produce an advancement. Through this project I was able to get a feeling for how codes are build up. It showed me how different pieces of code work together and produce a final outcome that becomes more powerful every step of the way.

I chose this project because this project not only because the context sounded interesting. Primarily I decided upon it because it combined something I already knew and things that I wanted to get to know better. In my past I already gained experience with microcontrollers which I thought was a good fundament to approach the project. Even though this proofed not to be entirely the case I think it was a good experience. It made me realize again that even though one might think he knows a large portion of a field of study there are always parts still to be discovered. On the other hand it also provided the possibility to learn more about radio communication and accelerometer handling. Furthermore I gained experience in getting accustomed to new tasks and in organizing project work. I think that this will help me to familiarize more quickly to new environments later on in my professional life. Also the broadening of my knowledge will help me when communicating with others.

In conclusion I can say that through this project I was able to improve my abilities, knowledge and overall experience in several fields.

# Annex

- Pins assignment
- Schema board
- Schema accelerometer

## *Pin assignement*

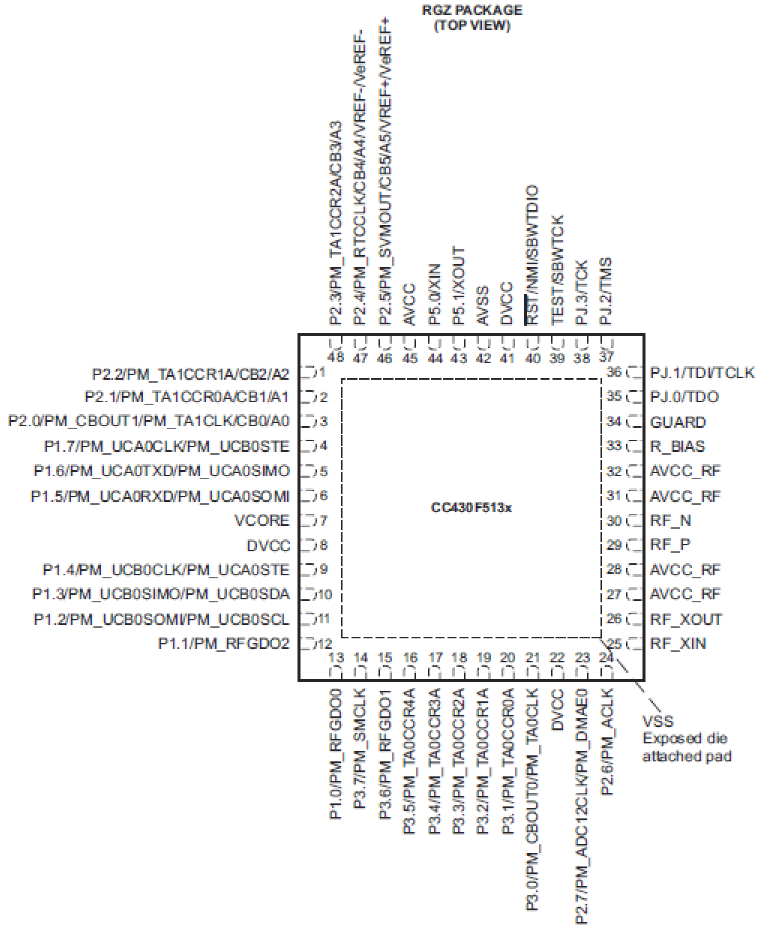



*Image 8: Pin Assignment*

## Schema Board

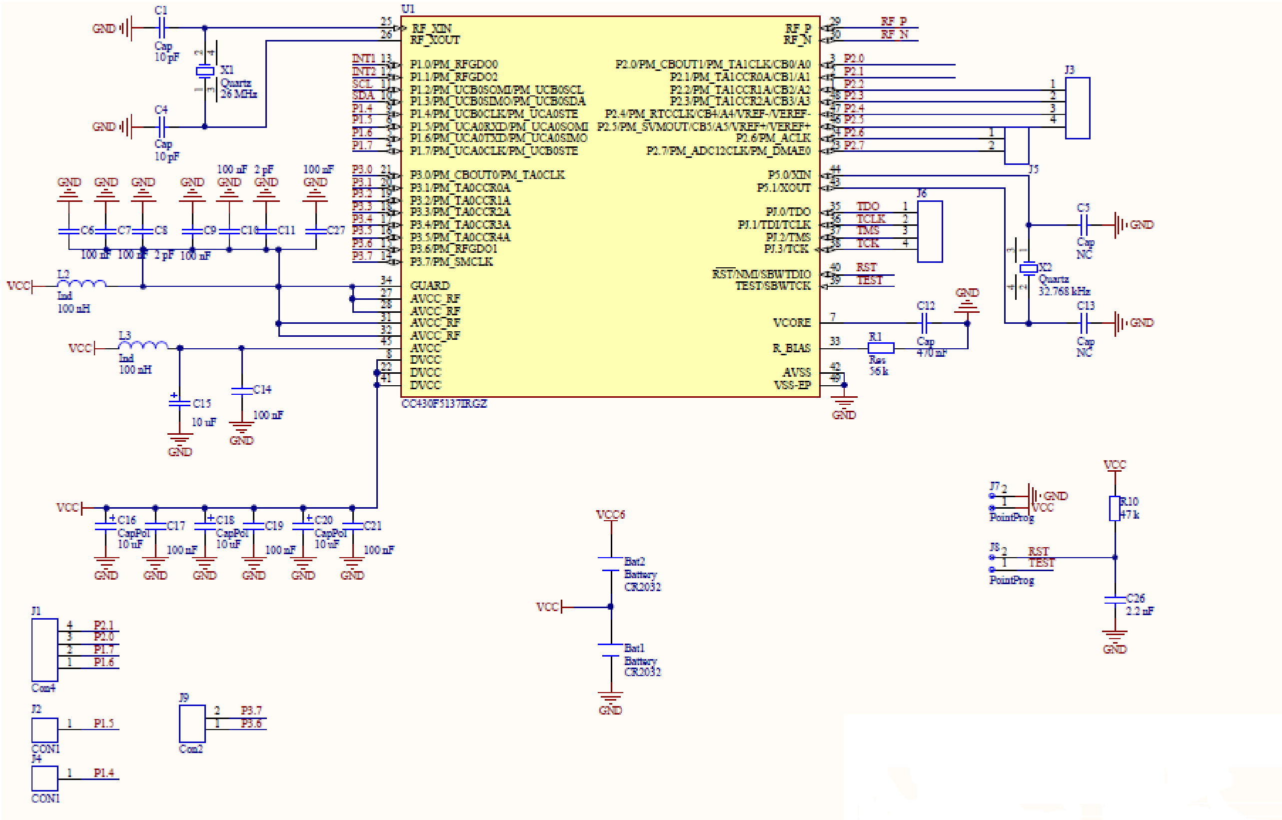


*Image 9: Schema Board*

## Schema Accelerometer

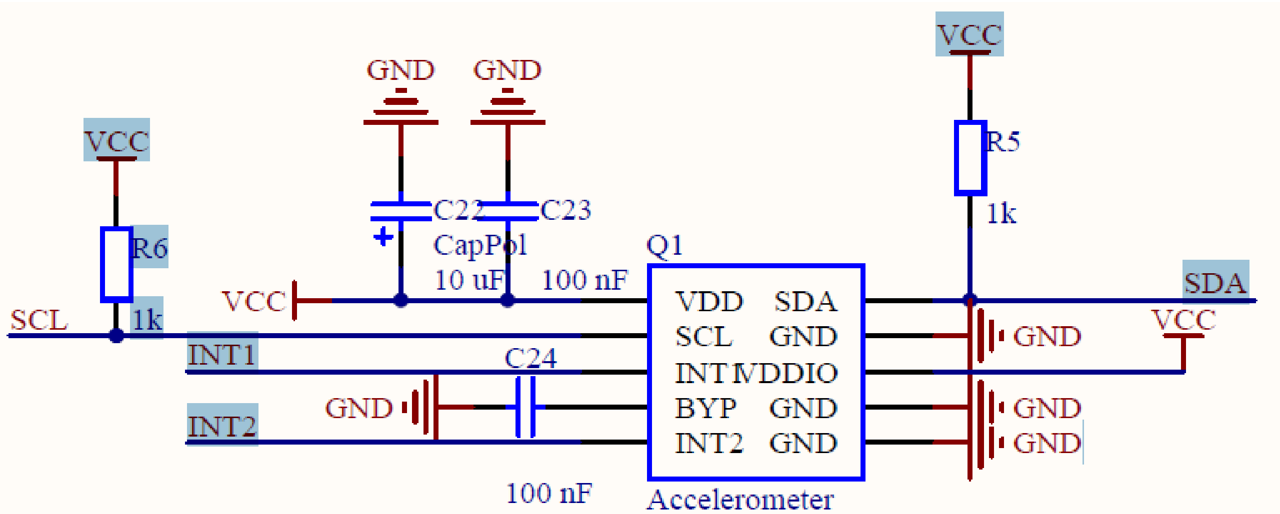



*Image 10: Schema Accelerometer*