Final Project: Heart Rate Sensor

Modified to Change Using CapSense

School of Engineering

Electrical and Computer Engineering Department

ECEG 721-61 Embedded Systems

Jamie Quinn & Lucia Rosado-Fournier

December 17th, 2020

Table of Contents

1.1 Objective

There are multiple objectives for this project. The first objective is to measure a simulated heart rate and send the data over Bluetooth Low Energy (BLE). The second objective is to change the original code given to us. The last objective is to gain a better understanding of embedded systems using modular design and abstraction.

1.2 Introduction

The heart rate is simulated by measuring an analog signal input. The signal is sent at a repeated interval, like a heart rate. The heart rate is measured by identifying these peaks in the signal that define the interval.

The BLE component of this lab will need to generate a notification every second to update the heart rate value to the client device. The BLE pioneer kit will act as a GATT server and send this signal over to the client.

For the original code, different modes will be implemented onto the device, with a watchdog timer (WDT) to act as the source of wakeup between the modes. The active mode handles the measurement of the signal and sends that information to the client device every second. The device will enter hibernate mode after being disconnected or timed out. It can then be woken up when the button SW2 is pressed.

For the updated code, it checks if the CapSense sensor is being touched. CapSense sensor is a capacitive touch sensing technology that measures changes in the capacitance between a plate (the sensor) and its environment to detect the presence of a finger on or near a touch surface. If the sensor is touched, the heart rate is simulated and sent to the device over BLE.

1.3 Component Requirements

1.3.1 Hardware Components

1. BLE Pioneer Kit
2. USB Cable
3. Apple or Android device

1.3.2 Software Components

1. PSoC Creator version 3.1 or higher
2. Cysmart 1.0 or higher
3. Cysmart on Apple or Android Device

1.4 Software

The code for the board is split into four files. The first file, debug.c, prints out debug messages onto the UART. This is helpful for debugging purposes and seeing what the code is doing as it's running. The second file, hrss.c, is the file that simulates the heart rate and the data produced here is sent over to the Apple or Android device over BLE. The next file, server.c, enables the BLE connection between the cell phone and the BLE Pioneer Kit. The file also puts the board into hibernate when the board is not connected to the phone for a few minutes and turns it back on when SW2 is pressed. Lastly, the main.c, shown below, first initializes everything and then enters a loop where it checks for the BLE connection to be established and then checks if the CapSense sensor is being touched. If touched, the heart rate is simulated and sent to the device over BLE. Compared to the original code, the WDT initialization and interrupt initialization was removed. To see the original and modified code, they are uploaded on GutHub.

C code of main.c

```
#include "hrss.h"

CYBLE_API_RESULT_T apiResult;
void AppCallBack(uint32 event, void* eventParam)
{
#ifdef DEBUG_OUT
    DebugOut(event, eventParam);
#endif

    switch(event)
    {
        case CYBLE_EVT_STACK_ON:
        case CYBLE_EVT_GAP_DEVICE_DISCONNECTED:
            heartRateSimulation = DISABLED;
            /* Put the device into discoverable mode so that remote can
search it. */
            StartAdvertisement();
            break;

        case CYBLE_EVT_GAP_DEVICE_CONNECTED:
            Advertising_LED_Write(LED_OFF);
            break;

        default:
            break;
    }
}

int main()
{
    CYBLE_STACK_LIB_VERSION_T stackVersion;

    CyGlobalIntEnable;

    UART_DEB_Start();               /* Start communication component */
    printf("BLE Heart Rate Sensor Example Project \r\n");

    Disconnect_LED_Write(LED_OFF);
    Advertising_LED_Write(LED_OFF);

    /* Start CYBLE component and register generic event handler */
    apiResult = CyBle_Start(AppCallBack);
    if(apiResult != CYBLE_ERROR_OK)
    {
        printf("CyBle_Start API Error: %x \r\n", apiResult);
    }
```

```
    apiResult = CyBle_GetStackLibraryVersion(&stackVersion);
    if(apiResult != CYBLE_ERROR_OK)
    {
        printf("CyBle_GetStackLibraryVersion API Error: 0x%x \r\n",
apiResult);
    }
    else
    {
        printf("Stack Version: %d.%d.%d.%d \r\n",
stackVersion.majorVersion,
            stackVersion.minorVersion, stackVersion.patch,
stackVersion.buildNumber);
    }

    /* Services initialization */
    HrsInit();
    CapSense_Start();

    //Initialize baselines
    CapSense_InitializeAllBaselines();


    while(1)
    {

        if(CyBle_GetState() == CYBLE_STATE_CONNECTED)
        {
            //Update baselines
            CapSense_UpdateEnabledBaselines();
            //Scan enabled sensors
            CapSense_ScanEnabledWidgets();
            //Wait for scanning to finish
            while(CapSense_IsBusy());

            //if touched, simulate heart rate and measure a battery level
            //and send results to the Client
            if(CapSense_CheckIsWidgetActive(CapSense_BUTTON0__BTN))
            {
                if(heartRateSimulation == ENABLED)
                {
                    SimulateHeartRate();
                }
            }
        }

        CyBle_ProcessEvents();
    }
}
```

1.5 Procedure

1 Build and program the BLE board.

2 On the cellphone with the Bluetooth enabled, connect to the board.

3 Go to the heart rate sensor on the phone and touch the upper part of the CapSense to see the heart rate change.

1.6 Observation

Also on GitHub is the footage, called "New Project - Made with Clipchamp", of the procedure outlined above. As shown in the footage, the heart rate changes when the upper part of the CapSense is touched. However, the heart rate changes a lot when pressed. This is because the CapSense is sensitive, so even a slight touch registers as several presses, changing the heart rate several times. Also in the footage is a video called "2020-12-06 15-23-12", which demonstrates the wake up from hibernate functionality when SW2 is pressed.

1.7 Summary

Due to time constraints, we were not able to refine this to have it change once upon a brief touch. Some other things we wanted to implement was the ability to change the heart rate by sliding the CapSense instead of touching one specific part of it. Sliding down to cause the heart rate to go down and sliding up to cause the heart rate to go up. Last thing we wanted to implement was the usage of an actual heart rate sensor connected to the BLE board.

Overall, the project challenged us to try something new and think out of the box to modify the existing heart rate sensor in a meaningful way. With more time, it would have been interesting to implement the changes outlined above.