University College of Southeast Norway
Department of Micro and Nano Systems Technology

Smart-clothing Platform for Elderly Care: Monitoring of Physical Activities, Physiological Status and Outdoor Localization in Real Time

Master's thesis

Author: He Niu

Supervisor: Tao Dong

A thesis
submitted in partial fulfillment of the requirements
for the degree of

Master of Engineering

University College of Southeast Norway

May 2017

University College of Southeast Norway

ABSTRACT

Faculty of Technology and Maritime

Department of Micro and Nano Systems Technology

Master of Engineering

Smart Clothing Platform for Elderly Care: Monitoring of Physical Activities, Physiological Status and Outdoor Localization in Real Time

by He Niu

The emerging research on smart clothing platform in health monitoring field has been a great concern both in academia and products industry. And with the problem of global population aging, the need for more services with higher quality in eldercare environment is poised to become significant. Advances in micro technologies, flexible sensors and related technologies make a new smart clothing can be developed to enhance the efficiency of eldercare, illness and unpredictable event prevention with monitoring and alarming method.

This master thesis provides a design of a smart-clothing platform based on embedded and wearable technology, which is a continuous monitoring to track the physical activities, physiological status and achieves automatic outdoor positioning for the elderly. Based on sensor network technology and Qt Creator development environment, the platform uses a greater number of sensor nodes to detect the different parts of the body to improve the detection accuracy. Front-end system uses STC12C5A60S2 microcontroller as the central node to receive the data of each sub-node, and through the wireless network data transfer to the PC for analysis. PC research algorithm to deal with the relevant data. Finally, the human-computer interface display and store results. Compared with traditional smart clothing, this system can provide a multifunctional specialization monitoring for elderly who need extra cares.

This thesis is organized as follows. Chapter 1is an introduction including the smart clothing concept, the review of previous works, motivations, and the objectives of this thesis is

also given in this chapter. Chapter 2 introduces system functions and the whole design scheme which covers both the hardware and software design. In chapter 3, the advanced relevant algorithms used for posture recognition is presented in order to provide the necessary knowledge, and the final algorithm used in this thesis is explained in this chapter. The chapter 4 and 5 respectively introduce the system hardware implementation and the software implementation. The system consists of wearable nodes (6 subnodes and 1 central node) and PC human-computer interaction interface, which are explained in detail in both chapters. Chapter 6 shows the experimental results of this smart-clothing platform and analyzes them in turn. On the basis of Chapter 6, Chapter 7 concludes the advantages and disadvantages of this system and gives the recommendations for future works. Additionally, the software source code of host computer and the lower computer are given in the appendix

ACKNOWLEDGEMENTS

This work of smart clothing platform is supported by Regionale Forksningfond Oslofjordfondet Projects: «Smart-tøy for eldreomsorg: Oppfølging av fysiske aktiviteter og overvåkning av fysiologisk status i sanntid» (proj. no. 260586).

I would like to express my sincere appreciation to my research advisor, Prof. Tao Dong for his time, patience and kind guidance during my master project. Thanks for his offering advice and encouragement on my research work, which profoundly influenced me in my scientific research.

My gratitude also goes to all of those at the department of Micro and Nano Systems Technology, who helped me learn the laboratory and BioMEMS techniques for making this study possible: Prof. Frank Karlsen, Prof. Erik Johannessen, Zekija Ramic and Thomas Marthinsen of University College of Southeast Norway. Thanks to Dr. Zhaochu Yang, Dr. Haakon Karlsen, Dr. Zhongyuan Shi, Dr. Nuno and Chen Xing, a group of wonderful human beings. Thanks for all of your advice, helpful comments and valuable suggestions to my research work and this master thesis.

The most special thanks to my parents from the bottom of my heart, who give me power and confidence with their unselfish dedication and support through all this lengthy process.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

PCB	Printed Circuit Board
GPRS	General Packet Radio Service
ECG	Electrocardiograph
FSB	Fabric Serial Bus
ESA	European Space Agency
SMA	Signal Magnitude Area
SMV	Signal Magnitude Vector
SVM	Support Vector Machine
SV	Support Vector
MEMS	Micro-electromechanical Systems
SPI	Single Program Initiation
I2C	Inter-Integrated Circuit
FIFO	First In, First Out
AD	Analog-to-digital
CPU	Central Processing Unit
SRAM	Static Random Access Memory
UART	Universal Asynchronous Receiver/Transmitter
PCA	Program Communication Area
MCU	Microprogrammed Control Unit
HTTPD	Hyper Text Transfer Protocol Daemon
SFD	Single Frequency Dialling

Chapter 1. INTRODUCTION

1.1 The concept of smart clothing and its development process

Smart clothing, also known as intelligent clothing, electronic textiles, smart garment and E-textiles, is described as a platform embedded with electronics and computing components with clothing functionality to solve the wearable problem so that it can provide interaction effect between human and itself by detecting signals, processing information, and triggering responses.

The concept of smart clothing has been initiated from the idea of wearable computer [1]. The earliest smart clothing used traditional fabrics and some common electronic components, such as sensors, resistors, diodes and chips, through the design of fabrics and clothing structures to attach electronic components to textiles or clothing. In the 1990s for military purposes, smart clothing had a booming development in the field of bio-medical and sportswear in America and European countries. In 1999, Royal Philips and Levi Strauss [2] successfully created the world's first commercial wearable electronic smart clothing. This smart clothing included entertainment (such as music) and communications (such as mobile phones) function, but it was not washable and collapsible. In addition, this garment was limited to manual production that could not meet the requirement of large-scale commercial production, resulting in a high cost of such garment.

Traditional electronic components like rigid printed circuit boards (PCB) implanting clothing can not be deformed at the same time with fabrics, which brings a significant discomfort ability. Besides, clothes are unable to easily cleaned without removing those electronic elements. Given all of this, scientists have tried to implant circuits into fabrics, so that they can become more wearable and washable with the same ability as fabric deformation and have a very high degree of integration. In the earlier stage, smart clothing paid attention to technical practicability. Most products remained primitive since technology in this field was underdeveloped. With the development of miniaturization of electronic devices and new-type sensors, developers realized they need change from a technological focus to a customer-oriented concern. Consequently, a lot of cooperative projects emerged between fashion and

electronics fields in short order.

The concept of smart clothing can be understood within the scope of functional clothing to achieve higher mobility and comfort, while technical functions keep increasing in recent years [1, 3]. The latest smart clothing products rely on nano technology for miniaturization of sensors, processors, memorizers and other electronic components combining with fibers to be made into special fibers, such as conductive fibers, and then using these special fibers woven into fabrics for producing clothing. Smart clothing made in this way, depending on the purpose of use, can add diverse electronic components into fibers thus having different functions. Cloth material has the function of induction by interweaving conductive fibers and general fibers, which can be made into flexible sensors or switches to improve clothing wearability greatly.

In 2000, the United States MIT Media Technology Lab [4] developed a smart clothing can access the Internet system, which marked a beginning of breaking away from the pure textile era of clothing design lasting for thousands of years, and into a new era of electronic textile. In 2002, the German semiconductor equipment manufacturer Infineon Technologies Company released a wearable “MP3 player jacket” in Japan. This jacket used a wearable electronic technology, which implanted a microelectronics chip in clothes fibers [5]. Three years later, Infineon Technologies improved the coat design by adding a row of the textile keyboard made from wire-connected conductive fabric on left-hand forearm sleeves. British Eleksen started working with Spyder company to sell sportswear which inserting ElekTex electronic fabric. ElekTex maintained the performance of traditional fabrics, for instance, abilities to soften, twist and scrub. The electronic smart fabric woven from ordinary conductive materials and clothing material wears out quickly and is brittle and uncomfortable when people wearing. Therefore, ProeTEX project combined natural fibers and fibers produced by nano-fabrication technology together to make clothing soft and the current conductivity greatly improved.

Although currently the development of conductive fibers and nanotechnology has made smart clothing remarkably increased in textile texture, appearance, and comfortable capability, before smart clothing become the consumption mainstream of electronic products and garments industry, it still need to overcome a lot of problems, such as flexibility, useful life, washability

and stability and cost is also one of the important factors.

1.1 A review of previous work

With in-depth studies of intelligent clothing, various functions of products have been developed and applied to all areas of life to improve people's quality of life.

1.1.1 Smart clothing for medical care

Taccini, N *et al* [6-8] proposed a smart underwear as shown in Fig. 1.1 for monitoring physiological and biomechanical variables affected by cardiovascular diseases with strain fabric sensors based on piezoresistive yarns and fabric electrodes realized with metal-based yarns, which uses GPRS wireless transmission system to transfer signals to the corresponding doctor to get the information about patients. In Fig. 1.1, breathing sensors, electrodes and piezoresistive sensors are placed in the front and side of smart underwear respectively and the clothing is sufficiently resilient to fit the body shape with integrating all conductive elements into the textile comfortably. As can be seen from Fig. 1.1 (c), four electrodes are used for the impedance pneumography.

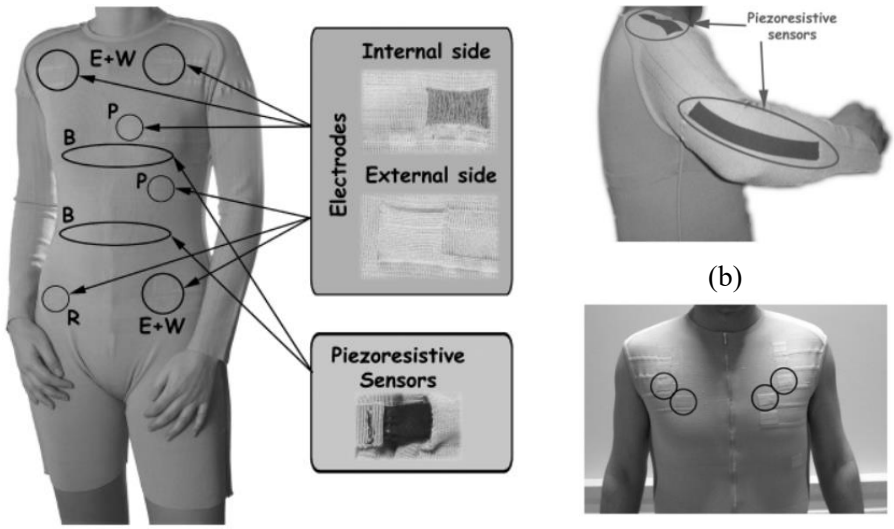


Figure 1.1 A prototype model for the smart underwear: (a) a front design, E+W, Einthoven and Wilson, R, reference, P, precordial leads, B, breathing sensors; (b) a profile design shows piezoresistive sensors; (c) electrode positions for impedance pneumography [8]

And some products, for example, Monbaby [9] and Owlet Baby Care [10] are smart clothes

for babies to help parents track baby sleep status, breathing movement, fall detection and send the information to a MonBaby app. A wearable device which can measure physiological and psychological data, including patients' ECG, blood pressure and Impedance Cardiogram (ICG) are designed by Srinivasan et al. [11]. These data are applied to analyze the user's emotional level using an embedded emotion classifier.

1.1.2 Smart clothing for recreation and entertainment

Sunwoo *et al* [12,13] presented a smart hooded jacket based on fabric serial bus (FSB) technology. Through conductive yarns to connect the hat and jacket and to stitch two chips, a light sensor, and an MP3 player, the MP3 player will play music when one person wears this hooded jacket and when it is removed, the music will stop. What's more, the volume of the music could change with the brightness of surrounding environment. While the wearer is in a dark environment, the music volume will automatically reduce and when the wearer comes out, the volume will return to normal.

The Hong Kong Polytechnic University has developed a dancing garment on the base of polypyrrole coated textile E-sensor that can detect the sensor signal placed on the elbow and knee of the dancing garment [14]. When the dancer changes her actions, output signals would let the audio system connecting with a computer to play corresponding music.

1.1.3 Smart clothing for physical culture and sports

Along with competitions increasing and people's attention to their own, the study of the smart clothing for physical culture and sports has gradually increased. The Hexoskin Smart Shirts [15] coming out in 2015 are a comfortable commercial product to help people to monitor their physical training with the information about heart rate, breathing rate, and activity intensity so that they are more suitable for professional athletes or body builders. Shirley Coyle *et al* [16] developed a novel sensor by carbon treatment after coating the conductive polypyrrole on the modified sponge to test the respiration rate during exercise. The sensor is connected to the processor via conductive yarns and communicating to the computer through a wireless transmission system for analysis and judgment. Every strength training and endurance training

can get a feedback instantly from this sensor, for example, recommending whether to continue using the same way to train or change training methods and if it is suitable for continuous training. It is helpful to physical health by improving breathing patterns, especially for high-pressure crowd.

1.1.4 Smart clothing for military use

Researchers at the Virginia Tech University have designed a wearable version of a giant textile, Sensornet, to detect noise [17]. The fabric, developed with support the US military, can be assembled on parachutes, tents or camouflage networks to receive data via built-in microphones, and the results can be transmitted to the outside or the corresponding system, which is used for the detection of military vehicles and tank movements. The intelligent military clothing embedded in biochemical and ultra-micro sensors has a lot of special features, including monitoring a number of physical functions (for example, blood pressure, heart rate, and body temperature) of soldiers, identifying the best hemostasis location to stem the bleeding and also resisting flames, chemical reagents and other battlefields to some extent.

In 2009, the European Space Agency (ESA) and Ohmatex, a Danish smart textile developer, have signed an agreement to jointly develop and test a smart sock device to depict the metabolic activity of the thigh muscles [18] with integrating near infrared sensors and the EMG electrodes based on textiles. The EMG sensor can be combined with socks by knitting machines, while near infrared sensors may be molded on socks. Ohmatex will use its rich experience in developing textile sensors to find the best solution. If development is successful, it will help to monitor and maintain the health of aerospace pilots during lunar prospecting missions through reducing the possibility of muscle damage.

1.2 Motivations for designing a novel smart-clothing platform

The problem of global population aging facing the world is an unprecedented situation. According to a report from World Health Organization (WHO) in 2015, it is estimated that by 2030, the number of people aged 60 years or over in the world will grow by 56 percent, to 1.4 billion, and by 2050, the global population of older persons will be more than double its size in

2015, reaching nearly 2.1 billion [19]. The situation in Asian countries is even worse with regard to the ever-increasing number of older people. The proportion of the aging population of Japan has exceeded 30%. By 2050, the level of population aging in many countries will rise to the same level, causing a series of problems around the world [20].

As the number of elderly people is increasing, need for more services with higher quality in a healthcare environment, such as the hospital and nursing homes, is poised to become significant. It is obviously that the elderly who can not look after themselves are in need of nurses or the family members paying more attention to them in order to prevent a situation like senile tumble or elderly vertigo. One possible countermeasure is to do a combination of both wireless sensor networks and mobile and wearable computing into health monitoring services, from which derives the concept of smart clothing (Fig. 1.2), including a few main components for biomedical use.

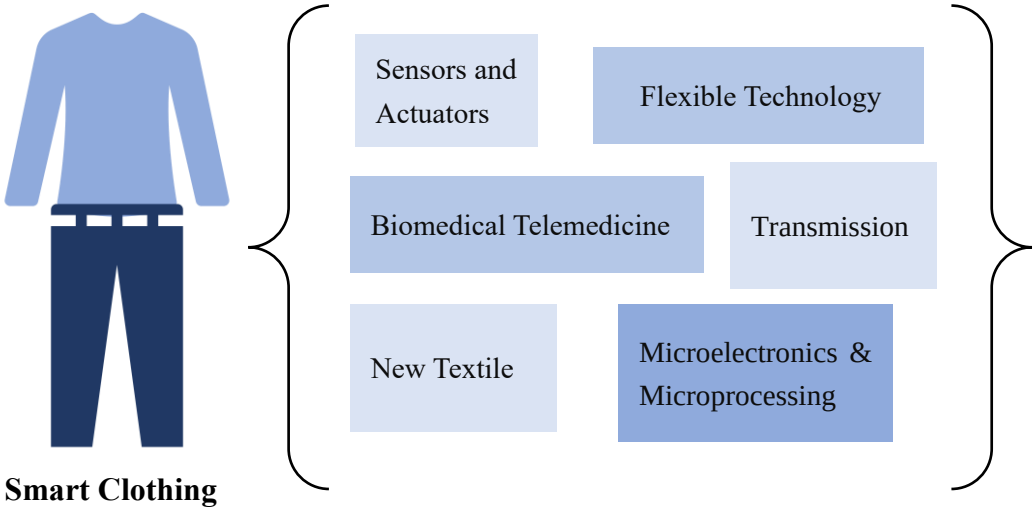


Figure 1.2 Main components in the biomedical smart clothing [21]

Now, we can see a requirement for elderly patients which is intended to improve comfort and also valuable for health monitoring with increasing efficiency and low cost during the health delivery. The elderly have high hopes for a higher quality of life, even when enduring a variety of chronic diseases. Thus, wearable monitoring systems and intelligent biomedical clothing can make sure a continuous monitoring to track the physical activities and achieve automatic outdoor location detection for the elderly. Besides, in order to enhance the efficiency of healthcare and illness prevention, this sustainable health-monitoring also need a real-time

physiological monitoring to provide the information about such as body temperature, heart rate, blood pressure or respiratory rate of the elderly.

The research and development of embedded and wearable technology have resulted in the emergence of biomedical intelligent clothing. Advances in miniaturization and new sensors open a door for the integration of electronic systems into light-small and compact construction [21]. The development of computing science is also key technical support for the attractive smart clothing.

The emerging research about smart clothing platform based on wearable technology has been a great concern both in academia and products industry. Practically, although there are many research achievements about health-monitoring, there are quite few similar smart clothing are able to meet the requirements for the elderly. Therefore, the service level of smart clothing should be further improved.

1.3 Main objectives of this thesis

According to the above motivations and previous research results, the main objectives of this thesis are:

- To put forward a design of smart-clothing platform to meet elderly people's demands in elderly care, which is a successive monitoring system to track the physical activities, monitor physiological parameters and realize outdoor orientation.
- To present a method after extracting the characteristic vector of the relevant gesture characteristics, which can classify and identify different gestures through a certain classification algorithm for analyzing and computing the signals of all sensors.
- To design a network communication model and implement a developing method for host computerized supervisory control system.

Chapter 2. SYSTEM FUNCTION & DESIGN SCHEME

2.1 Functions of smart-clothing platform

In this thesis, the design of the smart-clothing platform for the elderly can realize real-time health monitoring, physical activity monitoring and an outdoor positioning, including a wireless transmission function and a PC application software which benefits the nursing home and hospital in elderly care. For convalescent elderly patients, this platform can ameliorate the rehabilitation progress and monitor any complications in the early stages. In Fig.2.1, the functions of the smart-clothing platform as schematically with multiple sensors consist of three parts, monitoring the wearer's physical condition, tracking the outdoor position and monitoring of physiological status.

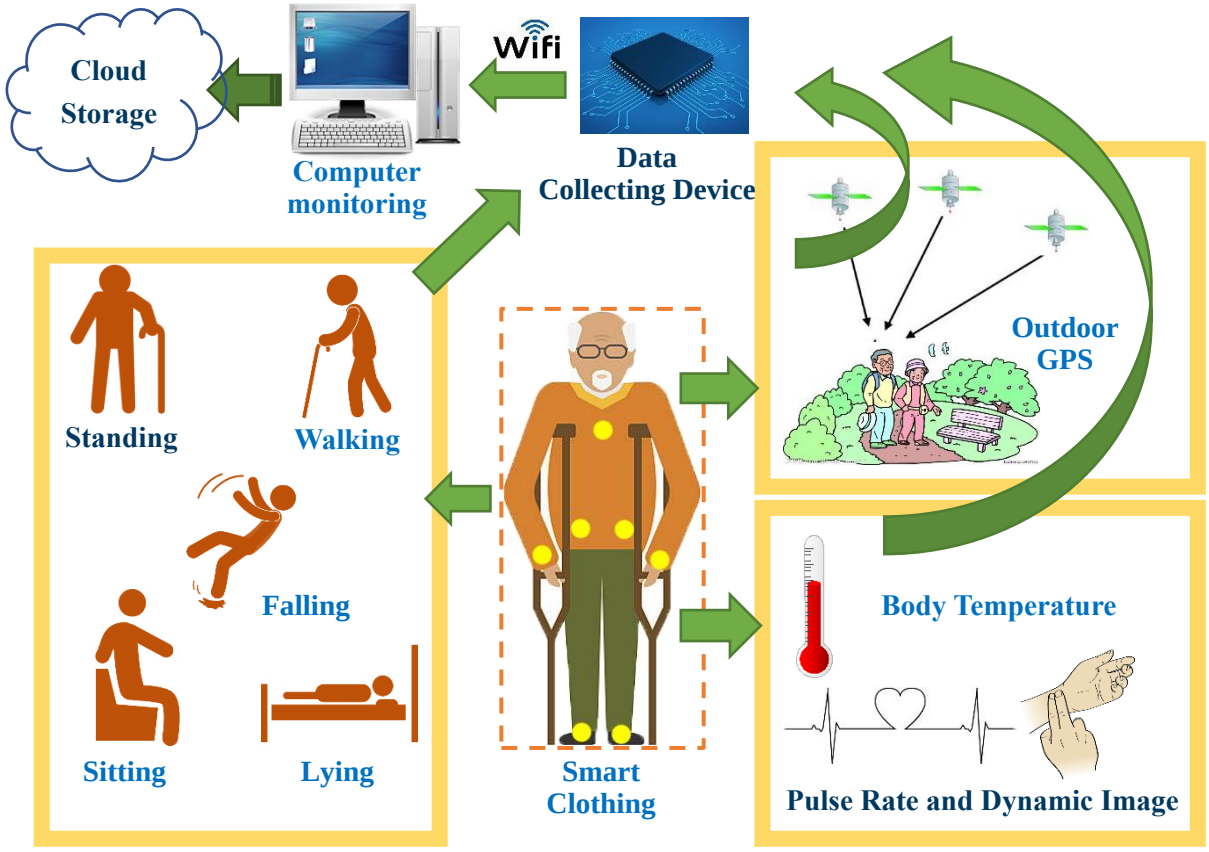


Figure 2.1 The functions of smart-clothing platform as schematically

The first part is physical activities monitoring which can detect five kinds of postures, standing, walking, falling, sitting and lying, to monitor the movement of the elderly to prevent

falling. The second part, outdoor localization that uses GPS method for preventing the loss of the elderly that can accurately obtain the position of the elderly around nursing homes or hospitals. And the third part is physiological status monitoring, otherwise known as a health-monitoring to sense body temperature and pulse rate, including pulse dynamic image for health care and illness prevention.

2.2 Whole system scheme

This paper studies a set of smart-clothing platform based on the wireless sensor network, which is used to detect the movement posture, physiological information and outdoor positioning of the elderly in daily activities.

2.2.1 Hardware and software design scheme

The whole system consists of hardware and software in two parts. The hardware part includes the sensor sub-node and the central node circuit design. It needs a reasonable choice to enlarge the filter circuit, extract the acceleration change information with a strong anti-interference, stable and reliable circuit design to guarantee the normal sending and receiving data. The software part includes the host computer software and the lower computer software design, in which the lower computer is responsible for data collection and sending and the host computer software for data processing and displaying the results on the human-computer interaction interface.

First, for the hardware part, it needs self-designed sensor node hardware circuit, which used to detect the human body lower limb movement acceleration, foot pressure, and buttocks force situation. It is required to minimize the volume to meet the wearable requirements while satisfying the system function. Figure 2.2 is the block diagram of the whole hardware system, including the power section, data acquisition and analysis section and wireless transmission, host receiving section.

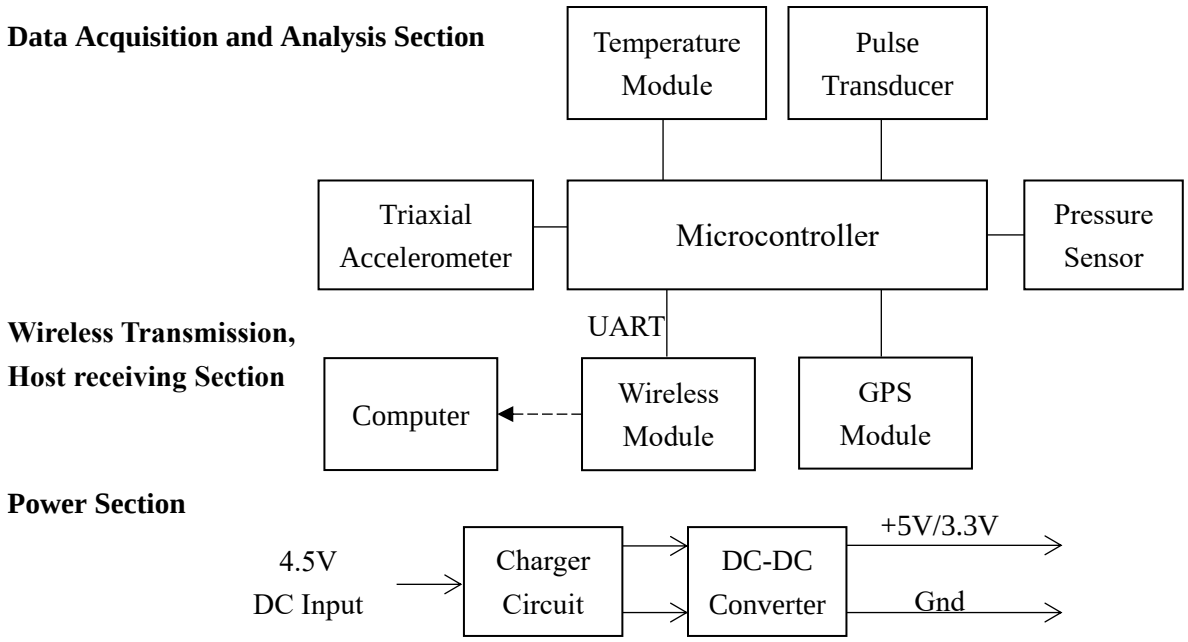


Figure 2.2 Block diagram of the overall hardware system design

Second, Software part needs to establish data acquisition and wireless transmission platform to collect the data from each sub-node sending to the microcontroller before forwarded to the PC, as shown in Fig. 2.3. On the computer, it should develop a human-computer interaction interface for real-time display of posture results, GPS positioning and monitoring basic physiological value with data storage function.

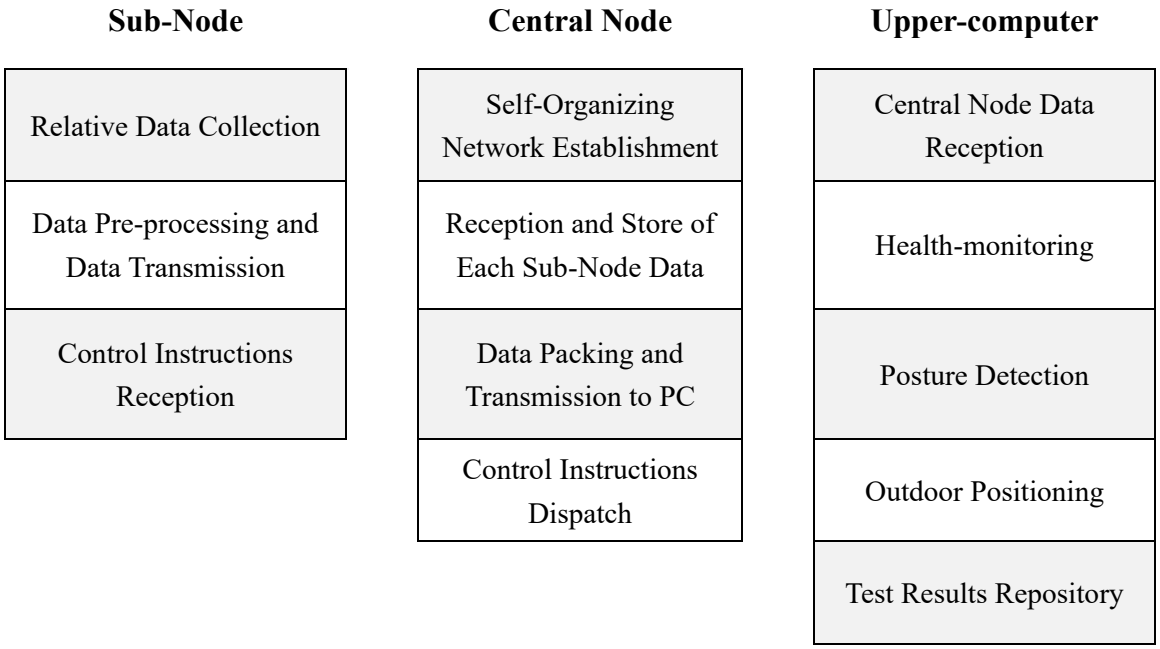


Figure 2.3 Function diagram of the overall software system design

2.2.2 Model establishment and algorithm implementation

Choosing multi-person to carry out a number of experiments about the designated motion. Firstly, the relevant data are collected and analyzed and then the data characteristics of the posture can be extracted for establishing the mathematical model to form a feasible attitude recognition method. This method is used to realize the recognition of the movement of the human body and the recognition of the different gestures.

According to the human body posture recognition, outdoor location and health monitoring method presented in this thesis, to design each program module and achieve the relevant identification method with data experiment to verify its effectiveness.

2.2.3 Wireless communication mode selection

Wearable systems may comprise various types of miniature sensors. The obtained measurements are communicated either via a wireless or a wired link to a central node, for example, a microcontroller board, which would, in turn, transmit these necessary signals to user interfaces or the monitoring center through one of the kinds of wireless communication modes [22].

Whether the data in a wireless communication system is safe and stable is essential. The main wireless transmission methods are radio frequency electromagnetic waves, infrared light and sound waves, and among them, the most commonly used is the radio frequency electromagnetic transmission mode. At present, there are ZigBee, Bluetooth, Wi-Fi and so on, which are widely used in medical and health care system. The following will compare the above three transmission technologies from the working frequency band, communication distance, transmission rate and power consumption to select the appropriate wireless communication technology, as shown in Table 2.1.

Table 2.1 Comparison of wireless communication technology between ZigBee, Bluetooth and Wi-Fi

	ZigBee	Bluetooth	Wi-Fi
Frequency range	2.4GHz/915MHz/868MHz	2.4GHz	2.4GHz/5.8GHz
Transmission rate	2.4GHz 250Kbps 915MHz 40Kbps 868MHz 20Kbps	1Mbps	2.4GHz 11Mbps 5.8GHz 54Mbps
Communication distance	10-75m	5-10m	100m
Power consumption	Transmitted power 1mw	Operating state 30mA Dormant state 0.3mA	250Ma~350mA
Device volume in a single network	256 devices, up to 65536	8 devices, up to 263	High capacity
Network topology	Star, tree and net shaped	Ad hoc, Scatternet	Large network

As can be seen from the Table 2.1, the biggest advantage of Wi-Fi is the high transmission speed, which can reach 54Mbps. While its effective distance is very long, but also compatible with a variety of existing 802.11 DSSS equipment. Therefore, this thesis chooses Wi-Fi to realize the data transmission of the smart-clothing platform system.

Chapter 3. METHOD AND THEORY

3.1 Classification of human activities posture

There are several basic static postures in the daily life: lying supine, left-lateral lying, right-lateral lying, lying prone, standing straight, standing prone, standing supine, sitting straight, sitting supine and sitting prone. Based on these basic human static posture, the dynamic transition between them forms the main dynamic model of human activity, including sitting down, lying down, walking, standing up and running.

3.1.1 Sit-stand-sit transformation

Standing up and sitting down activities, shown in Fig. 3.1, must depend on the balance of body center of gravity and the support and control of leg strength. The various stages of standing up consist of trunk lean forward movement, thigh angular displacement, trunk vertical upward direction of movement and standing straight. Under normal circumstances, the time required for body standing up is 1-5 seconds and increased with age. The elderly will last about 10 seconds. While sitting down process takes about 2 seconds and presents little relationships to age. And the extension of the knee position and the movement time in the vertical direction are closely related [23].

3.1.2 Walking and running

Walking is a very complex process that encompasses the results of the overall movement of different parts of the body. The way to walk is mainly affected by the following factors: foot state, body shape, walking purpose, physical health and mental state.

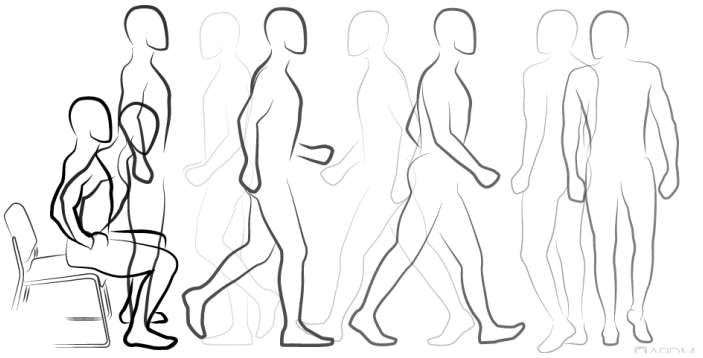


Figure 3.1 A typical sitting, standing and walking posture figure [24]

From Fig. 3.1, walking posture can be divided into several consecutive action events: the right heel striking the ground, right toe striking the ground, left heel lifting, left foot lifting forward, left heel striking the ground, left toe striking the ground, right heel lifting, right foot lifting forward.

Running gait is simpler than walking, but its movement rhythm and intensity is much higher than walking, during which the body torso is always moving forward.

3.1.3 Lying

The lying position usually includes supine, prone and lying on both sides, where the body is more or less horizontal and supported along its length by the underlying surface. In order to distinguish lying from standing and sitting posture, it takes into account the orientation of the sensor in relation to the direction of the gravitational acceleration [25].

3.1.4 Falling

In everyday life, the elderly often creep down or fall due to body balance ability declining gradually, causing serious injury and even paralysis and death. Falling is an involuntary loss of body balancing act. People usually cannot control the dumping action by themselves when they are falling, so falling generally only occurs in a short moment compared with the normal action. Because the body will be dumped in a flash, resulting in the movement of the center of gravity in the direction of tilt, the body action will produce an acceleration value.

3.2 Force analysis and gait cycle

In daily life, the most basic postures of the human body are walking, running, walking up

and down the stairs and other sports postures and lying, sitting, standing these static postures. The acceleration data collected by the acceleration sensors reflects the coherence of a series of actions of the human body, which also represents consecutive reactions of the force. During regular attitude motion of the human body, alternating and moving feet periodically, the force changes evenly so that the acceleration data in three directions (up and down, left and right, before and after) also show periodic fluctuations. Through the force analysis of the wear position of the moving equipment, the variation tendency of the acceleration data can be better understood; in turn, the current posture is able to be verified by the fluctuation of the data.

This thesis is from the force analysis of the movement of the leg position to extract and identify the characteristics of the human body. According to the force analysis at the time of taking steps, the fluctuation trend in the three axes of the acceleration data, which change with the transformation of the force, can be summarized. Finding the relatedness of the data of three axes at each time and further determining the data space of each step, and extracting the data characteristics of different postures can realize the recognition of unknown data.

When the human body is in a static state (such as standing, sitting, etc.), the acquisition modality is under force balance with the corresponding acceleration data almost unchanging. Basically, the corresponding data is its offset value and the value in the vertically downward direction is 9.8; horizontal front and back, horizontal left and right are 0. A regular movement state (such as walking and running), is produced by lower limbs stepping forward.

In the process of one stride, there are two key point-in-time under stress: moments of raising the foot and foot touching ground [26]. When one foot is raised, the human body overcomes gravity to do work with the upward force increasing. A lower limb lifts, at the same time, the forward force continues to drive for moving forward. When this lower limb touches the ground, footsteps suddenly stop, at this time, the force effect is obvious, especially in the horizontal direction. The role of force does not tend to balance until the center of gravity shifts. In the two point-in-time, the data acquisition equipment follows the movement of the body and the force effect is the most significant, which is gradually weakened in other time. The acceleration data would also fluctuate between peaks and valleys that generated in these two

moments. The following will be based on 3-axis direction (up and down, before and after, left and right) to further explain the changes in the force of one lower limb.

In the vertical up and down direction, when a person takes a step, overcoming the gravity to do work, the vertical upward force gradually increases to the maximum in the moment of lifting the foot, and now the acceleration sensor is in the overweight state with the measured data reaching the maximum. After the foot lifting, there is no upward force effect. Acceleration data become mitigating and the proportion of gravity action increases stage by stage until the lower limb reach its peak. After entering the settled down state, the lower limb is free to fall by gravity as the acceleration sensor is in the weightless state. The acceleration magnitude will be the minimum in the vertically downward direction and upswing in the processing of landing until the foot fall to the ground. Afterward, at the instant of foot touching the ground, the vertical upward acceleration is influenced by counter-acting force to gain constantly more than its offset value to the maximum. Later shifting of weight will result in it falls around to the offset once again.

In the horizontal front and rear direction, when a person takes a step, producing the forward force, as in the vertical direction of the situation, the acceleration sensor is in the weightless state in the horizontal rear direction with the data reaching a peak. From lifting the foot to setting foot down, the forward force of the human body may be durable and will not gradually decrease until the foot touch the ground, during which the acceleration sensor is mostly in a state similar to the overweight state backward. After setting it down, due to the counter-acting force of the ground, the sensor will produce a state similar to overweight forwards. And the date reaches to the maximum, at a later stable state, falling around to the offset once again.

When the human body is making a step, the direction of the force is vertical up and down and horizontal front and rear. In the horizontal left and right direction, the decomposition force is relatively small. The acceleration data will fluctuate around its mean and the different people's walking habits are distinct leading to different fluctuation range, but the peak size will not exceed the one in up and down, left and right direction.

A gait cycle includes stance phase and swing phase and the situation of the force in this

gait cycle is eventually reflected in the change of the acceleration magnitude. Through the analysis of the variation tendency of the acceleration signal, different signal variation of different postures can be identified scientifically, which in turn applies to different attitude recognition. Figure 3.2 is two samples about the acceleration data during a gait cycle. The data acquisition frequency is 100Hz. In this test, the horizontal forward is the x-axis positive direction; horizontal left is the y-axis positive direction; the vertical downward is the z-axis positive direction. The two zones labeled A and B represent the main force point in the process of a step.

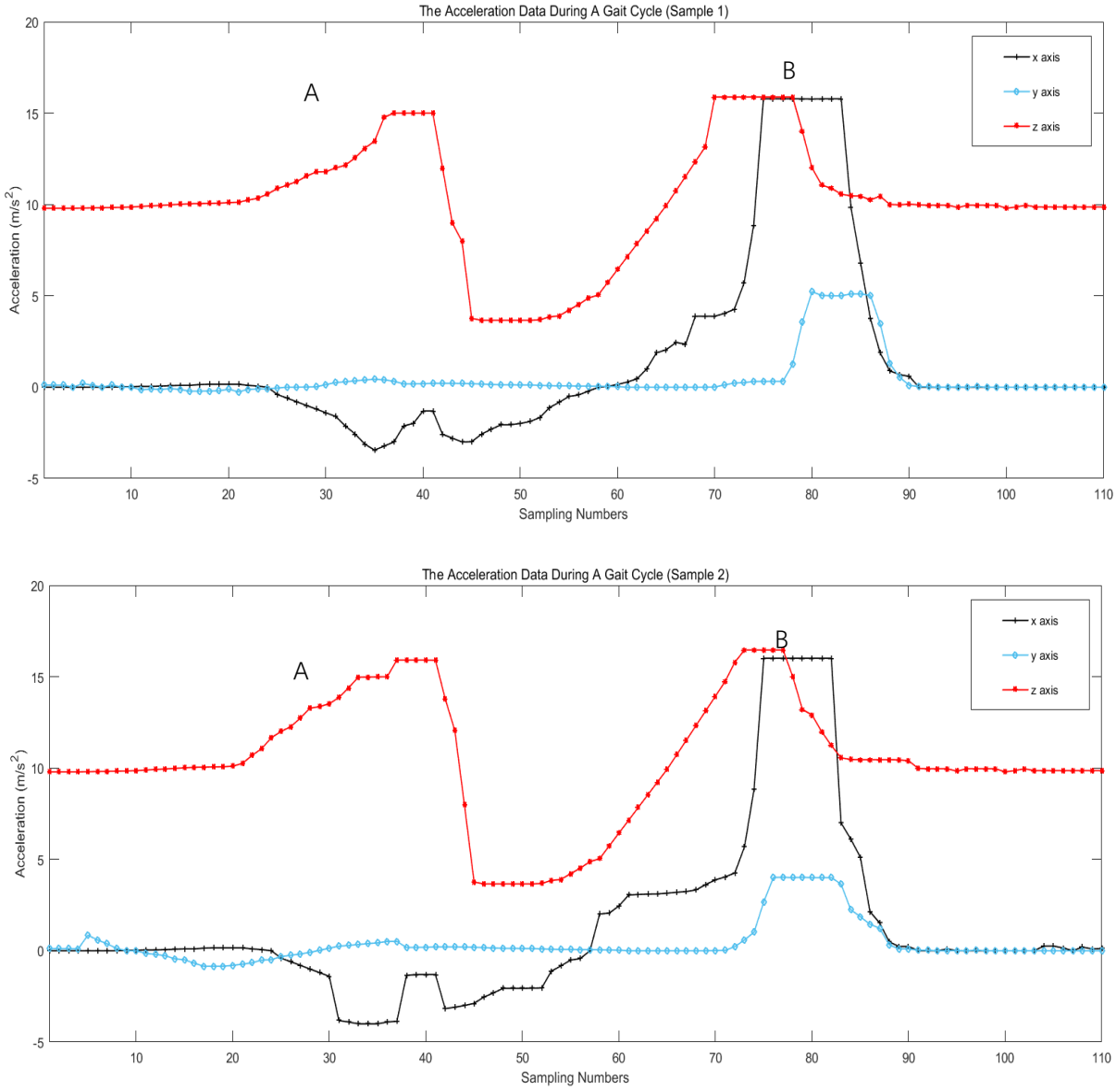


Figure 3.2 Two samples about the acceleration data during a gait cycle

From the Fig. 3.2, in the beginning, before zone A, the 3-axis acceleration data is almost unchanged. Z-axis (the vertical direction) is 9.8 m/s^2 and both of x-axis and y-axis (the horizontal direction) are 0 m/s^2. Afterward, from zone A to B, there are two obvious peaks and a valley in the z-axis direction and a distinct peak in the x-axis direction. The fluctuation of the y-axis is relatively small. Moreover, the peak position of x-axis direction and the second-time peak position of the z-axis are close to the maximum position of y-axis (no more than 10 data, about 0.1 seconds). After a fluctuation on a small scale, the three axes enter the relatively static state.

Combined with the force analysis of the human body during one step and many times experiments of lifting the foot and setting foot down, a gait cycle of one lower limb is divided into three stages: lifting, touchdown and alternating, as shown in Fig. 3.3, respectively.

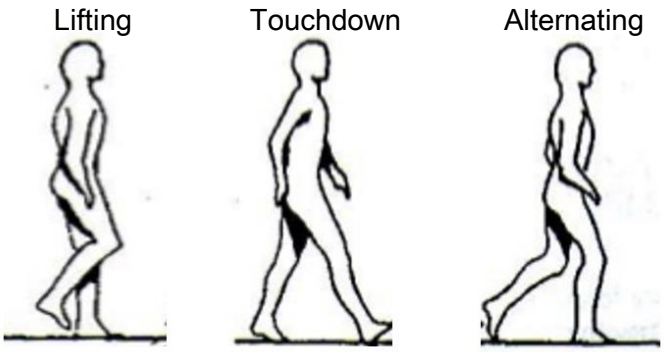


Figure 3.3 Three stages of a gait cycle of one lower limb [27]

- Lifting stage, in which the acceleration of y-axis increases first and then decreases and there are the first crest and an only valley corresponding to the moment of lifting the foot or setting the foot down respectively with the acceleration sensor in overweight or weightlessness. At the same time, the x-axis data are almost all below the offset 0, indicating that the acceleration sensor is pushed forward by the forward force. Z-axis data fluctuation is not obvious.
- Touchdown stage, in which y-axis data from the valley gradually increase to the second crest, while the x-axis data and z-axis data also increase, almost at the same time to reach the peak. This time corresponding to the moment down to the ground, the most intense moment of force. The second crest of the y-axis is different from the

first time because the second wave is relatively continuous for more than 10 data intervals, indicating the buffer process after landing.

- Alternating stage, in which the acceleration data gradually stabilized returning to a relatively static state because, after the violent fluctuations of the last stage, the center of gravity shifts and also with the foot alternation, the trunk, where the data collection equipment is placed, is relatively stable. When reaching the next cycle, the tendency of the above 3-axis data is repeated.

3.3 Fall detection technology

Fall detection technology is divided into three categories: video-based fall detection system, fall detection system based on the acoustic and fall detection system based on wearable sensors. Since the fall detection system based on wearable sensors is not limited by the location of the test, it is more suitable for smart-clothing platforms than video or acoustic systems. The fall detection system based on the wearable sensors embeds the microsensor into the wearable device. The system realizes monitoring the movement of the human body. When the movement parameters of the human body change, it is judged by the relative algorithm whether it has fallen. At present, the main methods of determining whether to fall are the threshold of judgment method and the use of pattern recognition method.

In contrast, the algorithm of the threshold of judgment is intuitive and the software cost is small. It is the most common method but the shortcomings are that the selection of threshold has a great influence on the experimental results. The general method is through empirical or experimental data to select the threshold. If choosing the empirical method, the accuracy of the threshold selection is unknown; and if choosing the experimental data for the threshold, there is a problem of verisimilitude, because the experimental data are basically obtained through healthy young people to simulate the fall. There will be differences in the physical and mental status of the fallers, and there are differences between conscious and unconscious falls, which affect the accuracy of the threshold selection. Therefore, how to select the threshold is one of the problems worthy of discussion. While the method based on pattern recognition is to deal

with the problem of fall detection by using the idea of complex problem simplification. By using some characteristic values such as SMA (Signal Magnitude Area), SMV (Signal Magnitude Vector) and spectral entropy [28], a statistical classification model is established, to classify the behavior of falls or not. This algorithm is highly adaptable, but the complexity is high. For the ordinary embedded microcontroller, there are more difficult.

3.3.1 The threshold of judgment method

A fall can generally be understood as standing or sitting posture converts into lying down posture with the human body in an unconscious situation. In the process of posture transformation, gravity will be the main factor to affect this motion process. The acceleration, velocity, and displacement of the object are changed during falling. Through certain means, the user's fall state is modeled and the acceleration characteristics of the object are extracted, from which the parameters can be used to analyzed the user's posture.

Compared with the normal movement of the human body (such as squatting, lying down, sitting down or bending, etc.), the falling action is more intense and the duration is usually 1-3 seconds. The body will be instantly dumped in a direction so that the body's center of gravity will follow it to move to the tilt direction. The result of such a body action is that there is a large acceleration change in the dumping direction. The three-axis acceleration sensor has real-time access to the acceleration data of the x, y, z three directions of the human body. Figure 3.4 indicates the acceleration data obtained during a fall backward and the data acquisition frequency is 100Hz, in which the horizontal forward is the x-axis positive direction; horizontal left is the y-axis positive direction; the vertical downward is the z-axis positive direction.

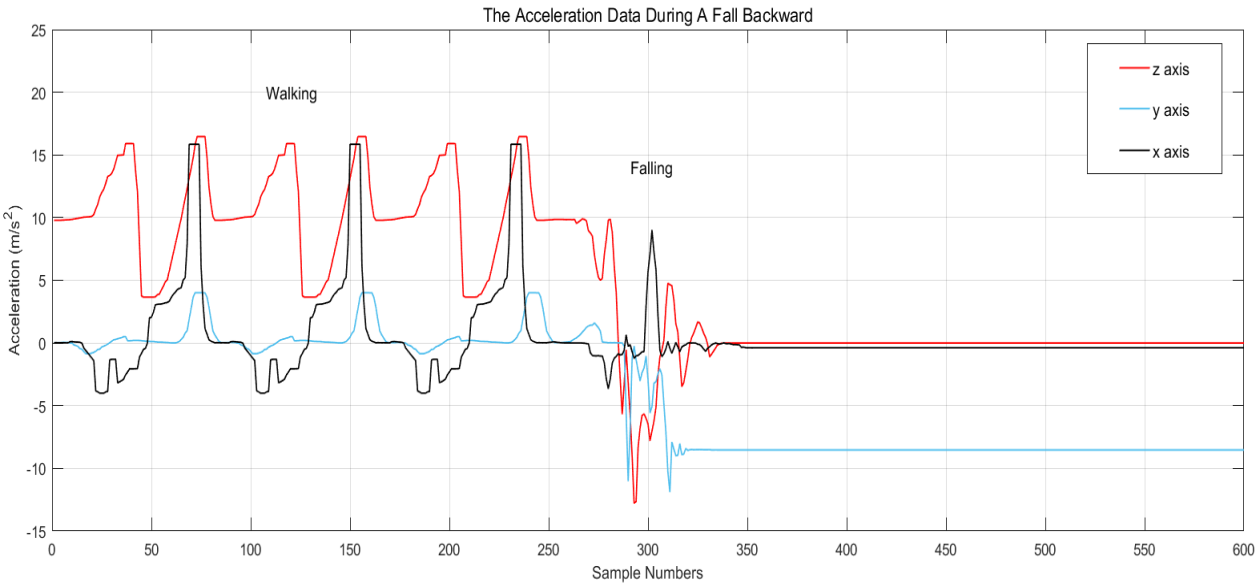


Figure 3.4 The norm of acceleration data during a fall backward

As can be seen from the figure 3.4, the fall takes place in the range of about 250-350. The interval after 350 is the relatively static state after falling. When the fall occurs, the data changes in each axis dramatically, and the peak deflection is large, especially the z-axis, in which the data offset is about 20. It is possible to determine whether or not a fall has occurred in comparison with the threshold values of the respective directions which set in advance before combining with other acceleration values, the angle information of the body or other data from different sensors.

In addition, when people stand or lie down, the foot pressure is different, which can assist in judging whether the body is in a falling state with getting from the pressure sensor placed in the sole of the foot.

3.3.2 The pattern recognition method

The fall detection algorithm based on the pattern recognition method mainly uses the latest SVM (Support Vector Machine) theory. This is a kind of method with strong theoretical basis, which is irrelevant to the definition of probability measure and large number theorem, so it is substantially different from the traditional statistical method.

SVM theory is first proposed by Vapnik et al. [29] in 1995. It is a new pattern recognition method developed on the basis of statistical learning theory, which shows many unique

advantages in solving small sample, nonlinear and high-dimensional pattern recognition problems in the pursuit of the optimal solution in the case of the existing information. A small number of support vectors determine the decision function of SVM, which is due to the small number, thus eliminating a lot of redundancy, and the decision also has a good robustness. The linear separability of the data means that a linear function can separate the existing data completely, otherwise, it is linear inseparable. The linear function is a point in the one-dimensional plane, a straight line for the two-dimensional plane and a plane in the three-dimensional space, and so on. Without paying attention to the dimension of the space, the hyperplane is representative of the generalized linear function.

Now we assume that the feature factor of the object to be identified is X_i (i=1,2,…,n). The corresponding ideal decision value is $Y_i \in \{-1,+1\}$. The optimal classification transcends the plane containing the weight vector w and the constant c with the equation:

$$wx + c = 0 \tag{1}$$

In order to classify all the samples correctly and to achieve the best results of the classification, it is usually necessary to use a sorting interval. The relationship between the geometric interval δ and the number of misclassifications of the sample ξ is as follows.

$$\xi < (\frac{2R}{\delta})^2, \tag{2}$$

where $R = \max(\| x_i \|), i = 1,2 \ldots n$, which means that the longest vector in all samples. It can be concluded that the larger the geometric interval is, the smaller the error upper bound is, so the goal is to maximize the geometric interval. Since the geometric interval is defined as

$$\delta = \frac{|g(x)|}{\|w\|}, \tag{3}$$

the maximized geometric interval is equivalent to minimizing $\| w \|$, that is, minimizing $\frac{\|w\|^2}{2}$.

The constraint condition for solving the classification is that the sample point must be on one side of the two classification surfaces and not in the middle of the two classification surfaces. The smallest interval 1 represents the two closest points in the sample, while the hyperplane must satisfy the following constraint equation.

$$\begin{cases} x_i \times w + c \geq +1, y_i = +1 \\ x_i \times w + c \leq -1, y_i = -1 \end{cases} \tag{4}$$

The equation (4) is transformed into (5) by quadratic programming. The equation (5) using a soft interval classifier, with a certain degree of fault tolerance. Hard classification thresholds need to add a slack variable ξ_i, which gives up the exact classification of these points, the classifier is a kind of loss, but it can get a greater geometric interval. In order to weigh this loss and the benefits, it is necessary to add the loss to the objective function and introduce the penalty factor S, which indicates the degree of emphasis on the loss of outlier sample points. The larger S is, the greater the loss of the outlier sample point to the objective function is, which means that the algorithm is more reluctant to give up these outliers, and the optimization problem becomes:

$$\begin{cases} \min \frac{1}{2} \| \mathrm{w} \|^2 + S \sum_{i=1} \xi_i , \\ subject\ \ to\ \ y_i(wx_i + c) \geq 1 - \xi_i, \xi_i \geq 0, i = 1,2 \ldots n, \end{cases} \tag{5}$$

Further using the Lagrangian multiplier α_i to simplify the above problem,

$$\begin{cases} \min L(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j (x_i', y_i') \\ \sum_{i=1}^{n} \alpha_i \, y_i = 0, \ 0 \leq \alpha_i \leq S_i \end{cases} \tag{6}$$

The kernel function realizes changing the data from linear indivisible to linearly separable by the spatial logical conversion of samples from low to high dimensional. $\mathrm{K}(x_i, y_i) = \Phi(x_i) \times \Phi(y_i)$. If $\mathrm{K}(x_i, y_i)$ meets the Mercer condition, there are:

$$\begin{cases} \min L(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathrm{K}(x_i, y_i) \\ \sum_{i=1}^{n} \alpha_i \, y_i = 0, \ 0 \leq \alpha_i \leq S_i \end{cases} \tag{7}$$

among them, the function of the kernel function is to accept two low-dimensional vectors. And the output is the vector inner product value in the high-dimensional space vector. Assume that the experiment selects the radial basis as a kernel function, as shown in equation (8):

$$\mathrm{K}(x_i, y_i) = \exp\left(-\frac{\|x_i - y_i\|^2}{\sigma^2}\right) \tag{8}$$

Suppose the optimal solution is $\alpha_i^* = (\alpha_1^*, \ldots, \alpha_n^*)$, then the problem should satisfy the equation (7). $\alpha_i^* \{y_i [w_0 x_i + c_0] - 1\} = 0, i = 1, \ldots, n$, where most of α_i are zero. And when α_i is not the value of 0, such a sample, Support Vector (SV) can finally get the optimal classification by virtue of the SV to obtain superplanar optimal classification. The decision function is as follows.

$$f(x) = \mathrm{sgn}(\sum_{i=1}^{n} \alpha_i^* y_i K(x_i, x) + c) \quad (9)$$

Finally, a large number of data samples of falling are collected by experiment. SVM training is carried out to calculate the superplanar optimal classification, and then the trained model is used to detect the actual samples to detect the fall events.

3.4 Posture detection algorithm

In order to accurately distinguish between falling events and daily behavior events, this smart-clothing platform coalesces information of one three-axis acceleration sensor and two pressure sensors to achieve real-time posture detection. The three-axis acceleration information collected by the sensor is analyzed, and the next step is determined using the two characteristic values of SMA [defined in (10)] and SMV [defined in (11)].

SMA is applied to discriminate between the periods of moving and stationary. SMA characterizes the degree of change of human movement, the greater the value that the more violent motion state changes [30].

$$\mathrm{SMA} = \frac{1}{t}[\int_0^t (|accX(t)| + |accY(t)| + |accZ(t)|)dt], \quad (10)$$

$$\mathrm{SMV} = \sqrt{accX_i{}^2 + accY_i^2 + accZ_i^2}, i = 1,2,\dots,n, \quad (11)$$

where $accX_i$ is the ith sample of acceleration value in the x-axis (similarly for $accY_i$ and $accZ_i$), in g (1g = 9.8Nm/s^2).

Because of the randomness of the time when falling happens, neither to predict the direction of the fall as the acceleration of each direction varies, and the axial deformation of the acceleration is more complicated, it is not conducive to the transplantation of the microprocessor. Hence, it is inappropriate to judge the fall by analyzing the acceleration of each direction separately. If using SMA and SMV to process the acceleration data with greater adaptability, it will ignore the spatial components of acceleration. SMA is the normalized region signal strength, which is the sum of the acceleration values within a short period of time. And the SMA in the event of strenuous movement and falls is much larger than the SMA in the case of normal minor movements. Based on this consideration, it can use the threshold algorithm to distinguish fall and another strenuous exercise from ordinary minor movement. In addition,

SMV is the signal intensity vector, real-time reflection the acceleration vector intensity of the current human body motion. In the vigorous exercise of human body, the signal intensity vector is larger, while in the slight movement, it is small. The same can be set by the threshold to is distinguish strenuous exercise from ordinary minor movements.

In addition, due to the impact of body gravity, when the body is in different daily activities, the soles of the feet are under greater pressure, and when the human body in the sitting or the lying state, the body's gravity is borne by the other parts, such as the buttocks. In this way, it can collect the pressure sensor data on the foot and buttocks to assist the posture detection, to distinguish between the sitting, lying and standing, which are static states. The use of a serial debugging assistant to see the return of the pressure value can easily determine the judgment threshold, whose effect is very obvious.

In conclusion, the posture detection algorithm of this smart-clothing platform, as shown in Fig. 3.5, first determines the SMA. In the SMA calculation, the calculated period is 0.5s, and the calculated time window length is 1s. When the SMA exceeds the threshold TH_1, it is assumed that the old person is in a walking or falling state and then further detects the SMV. When the SMA exceeds the threshold value TH_2, it is considered that a fall event occurs. If the SMA is less than the threshold TH_1, the old person is considered to be in a static action (sitting, lying or standing). At this time to extract the foot pressure ($Pressure_1$) and compare with the threshold TH_3, if $Pressure_1 > TH_3$, the posture is standing and if not, it is sitting or lying. Then using the buttock pressure ($Pressure_2$) to distinguish between lying and sitting in two cases. The four thresholds are shown in the following table.

Table 3.1 Four thresholds used in the posture detection algorithm (1g = 9.8Nm/s^2,1kg=9.8N)

TH_1	TH_2	TH_3	TH_4
0.8g	2.6g	35kg	30kg

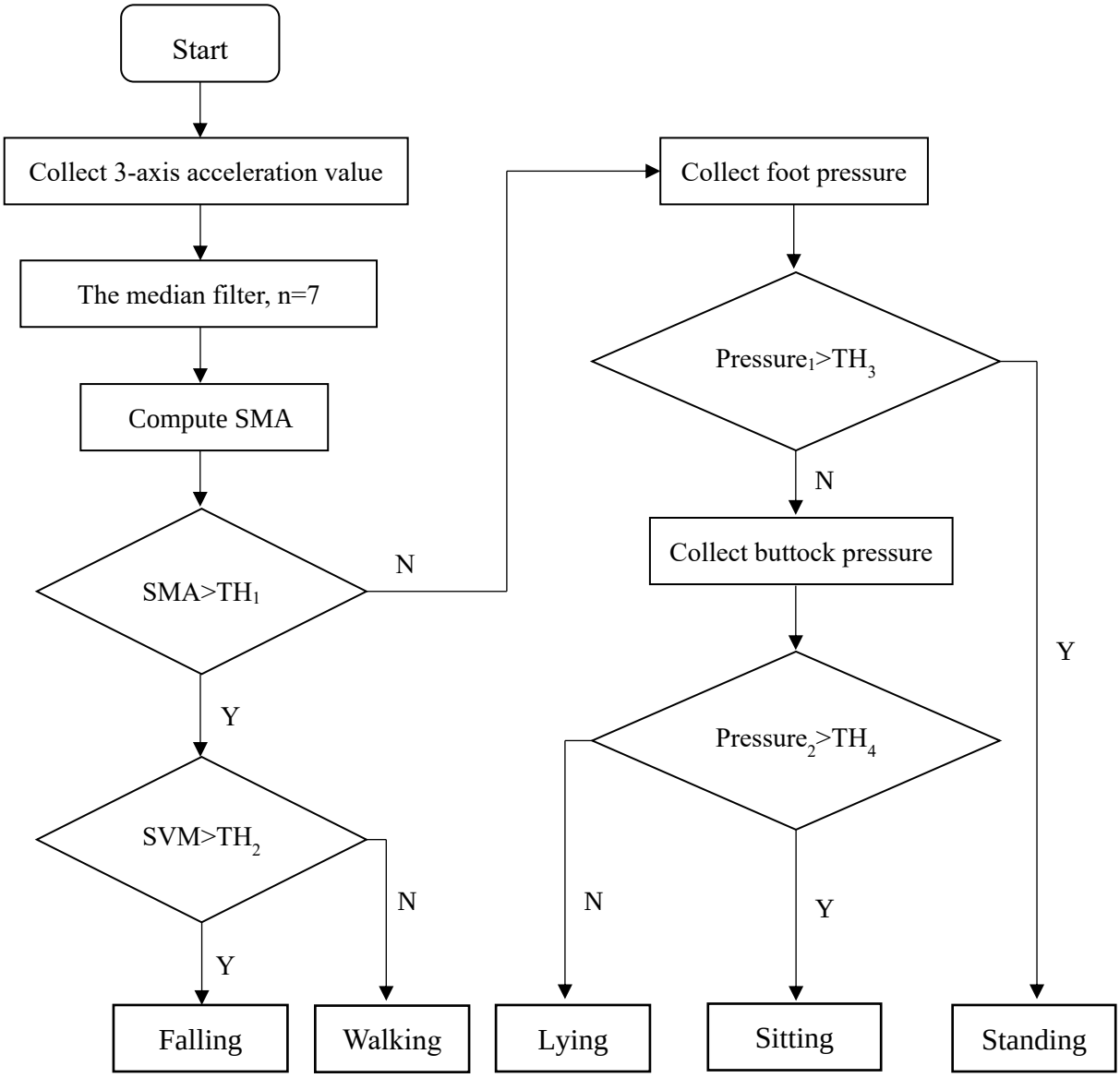
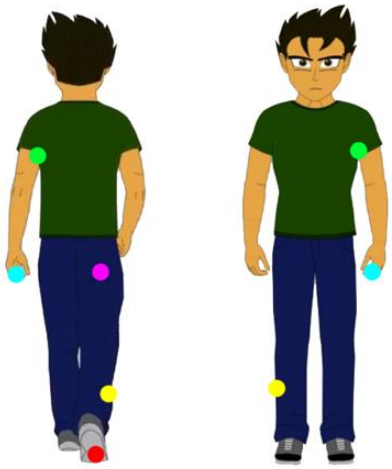
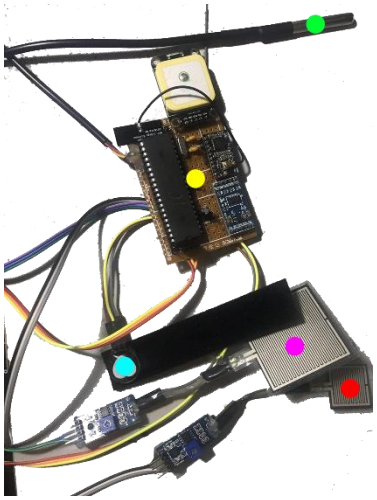


Figure 3.5 Posture detection algorithm flow diagram

Chapter 4. HARDWARE IMPLEMENTATION





System	Position	Component
Physical activities monitoring		3-axis Accelerometer, ADXL345
		Thin Film Pressure Sensor
		Thin Film Pressure Sensor
Outdoor localization		GPS Module, UBLOX-6M
Physiological status monitoring		Temperature Sensor, DS18B20
		Pulse Sensor
Master module		Microcontroller, STC12C5A60S2
		Wi-Fi module, USR-C215
		Power Supply, 4.5V Battery

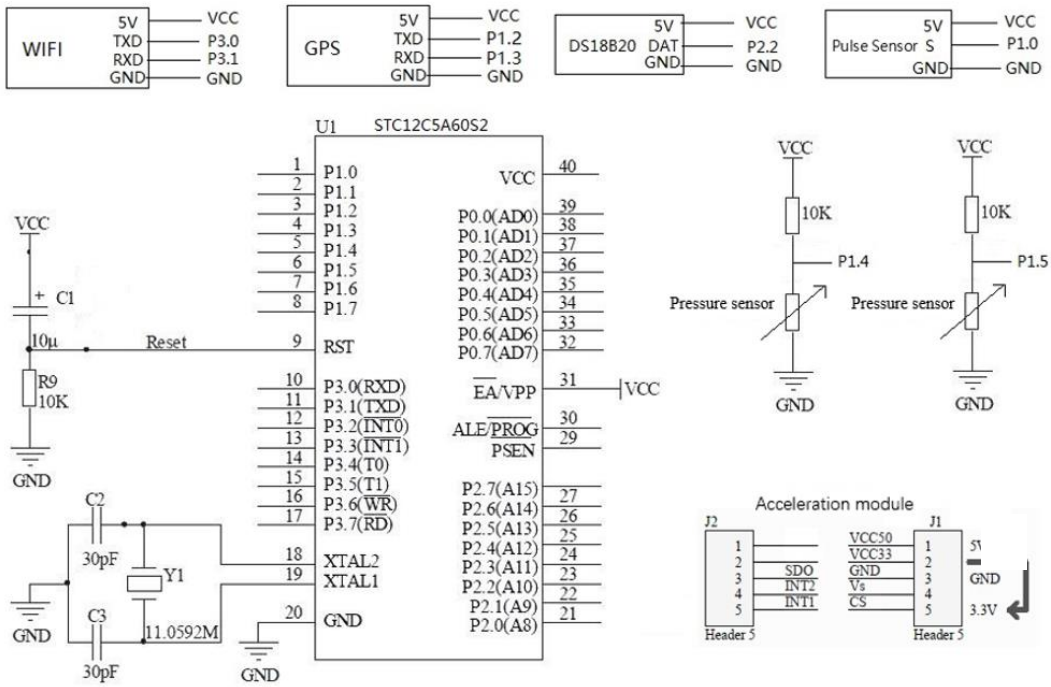


Figure 4.1 Smart-clothing hardware implementation: main components and circuit diagram

The hardware architecture of the design, as shown in Fig. 4.1, includes a central processor module, a 3-axis accelerometer, a power module and a signal transmission module on the lower leg, a single foot and hip pressure acquisition module, a body temperature acquisition module, and a pulse module, which can be divided into four sections: physical activities monitoring, outdoor localization, physiological status monitoring and master module. All the main components will be described in detail in this chapter.

4.1 Physical activities monitoring

4.1.1 ADXL345 3-axis accelerometer

ADXL345 (Fig. 4.2) produced by ADI Company in 2008 with MEMS technology with SPI and I2C digital output function is a three-axis accelerometer, with small, lightweight, ultra-low power, variable range, high resolution, and other features [31].

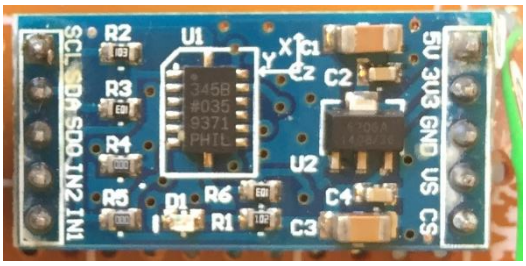


Figure 4.2 The real product of ADXL345 3-axis accelerometer

ADXL345 is only 3 mm × 5 mm × 1 mm in size and the face size is equivalent to 1/3 of the little thumbnail cover. At the typical voltage, V_S = 2.5 V, the power consumption is approximately 25 ~ 130 μA, compared with the previous use of analog output products ADXL330, the typical power consumption value decreasing about 70 ~ 175 μA. The maximum range of ADVL345 is up to ± 16 g, and the other options, including ± 2, ± 4, ± 8g range, which can adopt a fixed 4 mg/LSB resolution mode able to measuring the 0.25° inclination change.

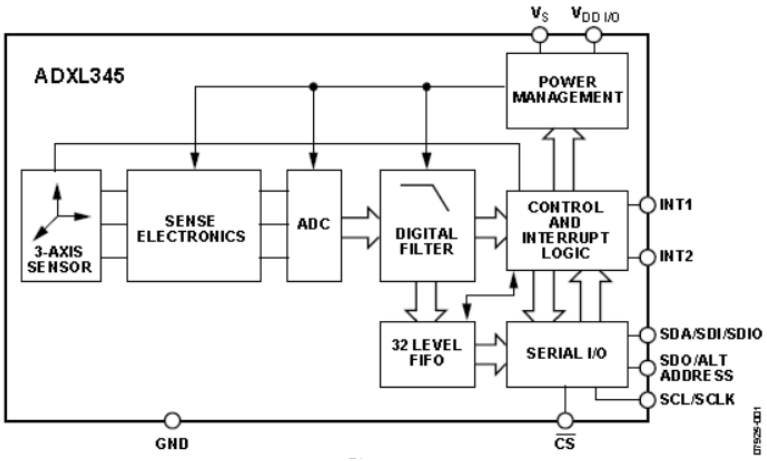


Figure 4.3 Function block diagram of ADXL345 [31]

Fig. 4.3 is the function block diagram of ADXL345. It provides some special motion detection function that can detect whether the object is in motion, and be sensitive to whether a certain axial acceleration is beyond the user-defined threshold, and also can detect whether the object is falling. In addition, a 32-level FIFO buffer register is integrated for cache data reducing the burden on the processor. The ADXL345 measures static gravitational acceleration in tilt-sensitive applications and dynamic acceleration in motion and even vibrating environments, making it ideal for mobile devices. It is expected to be widely used in mobile phones, games and positioning devices, miniature navigation devices, hard disk protection, sports, and fitness equipment, digital cameras and other products.

The principle of measuring the acceleration of ADXL345 is the sensor of this accelerometer that is a polysilicon surface micromachined structure placed at the top of the wafer. The polysilicon springs hang over the structure of the wafer surface to provide power resistance on account of applied acceleration. The differential capacitor consists of independent fixed plates and junction plates that can measure the structural deflection. Acceleration causes the inertia mass to be deflected and the differential capacitance unbalanced, so that the amplitude of the sensor output is proportional to the acceleration.

The working principle of the module is first to sense the magnitude of the acceleration by the front-end sensing device and change from an inductive electrical signal to a recognizable electrical signal, which is an analog signal. The ADXL345 integrates an AD converter that digitizes this analog signal, and as we know the digital signal is always represented in the form

of a complement in the computer system. In this case, the AD converter outputs 16-bit binary complement. After filtering through the digital filter, it accesses the 32-level FIFO under the control of interrupt logic unit, through the serial interface to read the data. By receiving read and write commands from the serial port to achieve ADXL345 control commands, which is mainly on the operation of the register.

4.1.2 RFP-602 thin film pressure sensor

RFP-602 thin film pressure sensor (Fig. 4.4) can be a static and dynamic measurement of any contact surface pressure. With multi-channel pressure collector, the user can see the intuitive, two-dimensional image of the real-time display of the pressure values, while the entire measurement process can be stored. The user can view and analyze the measurement records at any time. The pressure detection system should consist of hardware and software, including PC-based A/D conversion circuitry and reusable sensors; software is for pressure display and data analysis.

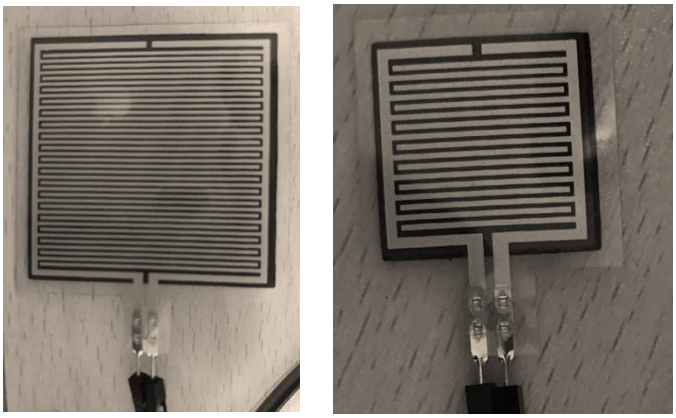


Figure 4.4 RFP-602 installed on the buttock (left) and in the sole of the foot(right)

The thickness of RFP-602 is only 0.1mm-0.2mm. The flexibility is very good, so as to create better conditions for measuring the pressure between the various contact surfaces. This is an economical, efficient, accurate, fast, intuitive and convenient pressure distribution measurement tool compared to conventional measurement methods before. The standard RFP pressure sensor consists of two very thin polyester films with the inner surface laying of conductors and semiconductors. When the external force acts on the sensing point, the semiconductor resistance will be proportional to the external force changes. while the pressure

is zero, resistance is maximum; the greater the pressure, the smaller the resistance.

In general, the sensor can directly collect the resistance. According to the resistance value to calculate the corresponding force range, so only an ohmmeter is needed for the measuring circuit. However, in the actual test process, because the direct acquisition of resistance equipment is unsuitable for integration, the general use is voltage acquisition card. That is, the resistance signal is converted to a voltage before the signal acquisition, then according to the test voltage to calculate the corresponding force size.

Table 4.1 Relationship between sensor loading and AD value of RFP-602

Weight	AD value
0kg	0
5kg	67
10kg	141
25kg	612
50kg	1021

The maximum operating current is 5mA. And RFP resistor-load response curve approximately obeys the inverse power law characteristic. Table 4.1 shows the unified relationship after calibrating between pressure value of sensor loading and AD value of two RFP-602 with different resistance values.

4.2 Outdoor localization

4.2.1 UBLOX 6M GPS module

UBLOX 6M GPS module is a high-precision, high-performance, and miniaturized module, with extremely high sensitivity to greatly expand the positioning coverage. In the place where ordinary GPS receiver module is unable to orientate, such as the narrow alleys, thick forest, UBLOX 6M also can realize high accuracy locating. The input voltage is 3.3-5.5V and the power consumption in normal mode is 50mA. The pin description and electrical connection are shown in Table 4.2.

Figure 4.5 The real product of UBLOX 6M GPS module

Table 4.2 Pin description and electrical connection of UBLOX 6M

Number	Pin definition	Description
1	VCC	Power supply pin; 3.3V~5.5V
2	GND	Power Ground
3	RXD	Module serial ports reception pin
4	TXD	Module serial ports transmission pin
5	PPS	Clock pulse output

4.3 Physiological status monitoring

4.3.1 DS18B20 temperature sensor

The temperature sensor is made of DALLAS company's DS18B20 networkable digital temperature sensor chip package, as shown in Fig. 3.9, with the characteristics of wear resistance, small size, easily using, packaging in various forms, being suitable for a variety of small space equipment, digital temperature measurement, and control field. Table 3.4 indicates the pin description of this module.

Table 4.3 Pin description of DS18B20 temperature module [33]

Number	Pin definition	Description
1	GND	Power Ground
2	I/O	Digital signal input/output
3	VDD	Power supply pin;3.0~5.5V

Figure 4.6 DS18B20 temperature sensor (left) and pulse sensor (right)

DS18B20 temperature measurement principle is that the oscillation frequency of low-temperature coefficient crystal affected by the temperature is very small. Pulse signal used to generate a fixed frequency transfer to the counter No.1. The high-temperature-coefficient crystal, whose oscillation rate changes significantly with the temperature changing. The resulting signal is the pulse input of counter No.2. Counter No.1 performs subtraction counting for the pulse signal generated by the low-temperature coefficient crystal. When the preset value of the counter No.1 is reduced to 0, the value of the temperature register adds one and the preset of the counter No.1 will be reloaded to restart counting pulse signals generated by the low-temperature coefficient crystal until the counter No.2 counts up to 0 to stop the accumulation of the temperature register value. At this time, the value in the temperature register is the measured temperature.

4.3.2 Pulse Sensor

Pulse sensor (Fig. 4.6) is a light-reflective analog sensor used for pulse heart rate measurement. Users wear it on the finger, earlobe, etc., with the wire connection, which can transmit collected analog signals to the microcontroller used to convert digital signals. After a simple calculation of the microcontroller, it can get heart rate values, in addition, uploaded pulse waveform is able to display on the computer.

There are mainly three traditional pulse measurement methods. First, extraction of the pulse signal from ECG signal; second, calculating the pulse rate using fluctuations measured by pressure sensor during the measurement of blood pressure; third, photoelectric

plethysmography. The first two methods to extract the signal will limit the user activities, and long-term use will increase the patient's physical and psychological discomfort. While photoelectric plethysmography is one of the most common methods of monitoring the measurement, with a simple method, easy to wear, reliable and safe features.

The basic theory of photoelectric plethysmography is to utilize different luminousness caused by body tissue during vascular pulsation for pulse measurement. The sensor used is composed of two parts: a light source and a photoelectric converter, fixed by a strap or clip on a patient's finger or earlobe. A light source generally chooses the light-emitting diode with a 500nm~700nm wavelength which is selective about the oxygen in arterial blood and hemoglobin. When the light beam transmits human peripheral blood vessels, the light transmission rate changes due to volume changes caused by artery pulse. At this point, the light reflected by body tissues received by photoelectric converter changes into the electrical signal to amplify. Since the pulse is a signal that changes periodically with the beat of the heart, and the arterial volume changes cyclically, the electrical signal change cycle of the photoelectric converter is the pulse rate.

Fig. 4.7 indicates the structure flowchart of pulse sensor. This sensor uses a green LED with 515nm peak wavelength, model AM2520. While optical receiver chooses the APDS-9008, which is an ambient light sensor with 565nm perceived peak wavelength. A low-pass filter and an amplifier made of op amp MCO6001 are utilized after the photoreceptor to amplify the signal by 331 times.

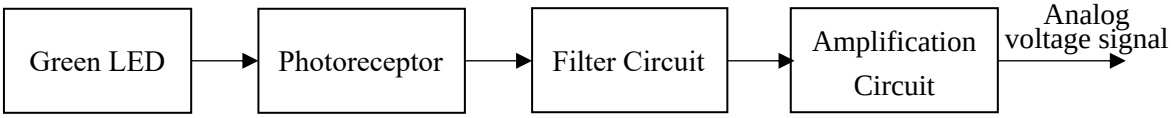


Figure 4.7 The pulse sensor structure flowchart

4.4 Master module

4.4.1 STC12C5A60S2 microcontroller

The microcontroller is the core of the entire front-end system to complete all the things in the front-end system scheduling. Because the smart-clothing platform is a portable product, the

microcontroller needs a characteristic of low power consumption. According to requirements analysis, the components associated with the MCU in the design are acceleration acquisition, pressure acquisition, temperature acquisition, pulse acquisition, wireless transmission modules and so on, so need to choose an interface communication mode of the microcontroller meeting the requirements.

STC12C5A60S2, a single-chip microcomputer produced by Hongjing Technology Company is a single-clock microcontroller with the machine cycle (1T). It is a high-speed, low power, ultra-strong anti-interference and new generation of 8051 single-chip. The instruction code is fully compatible with the traditional 8051, but faster than it 8 to 12 times faster. This microcontroller contains a CPU, program storage, SRAM, timing counter, UART serial port, I/O port, high speed A/D converter, SPI port, PCA, watchdog, an on-chip clock oscillator, an external oscillation circuit and so on.

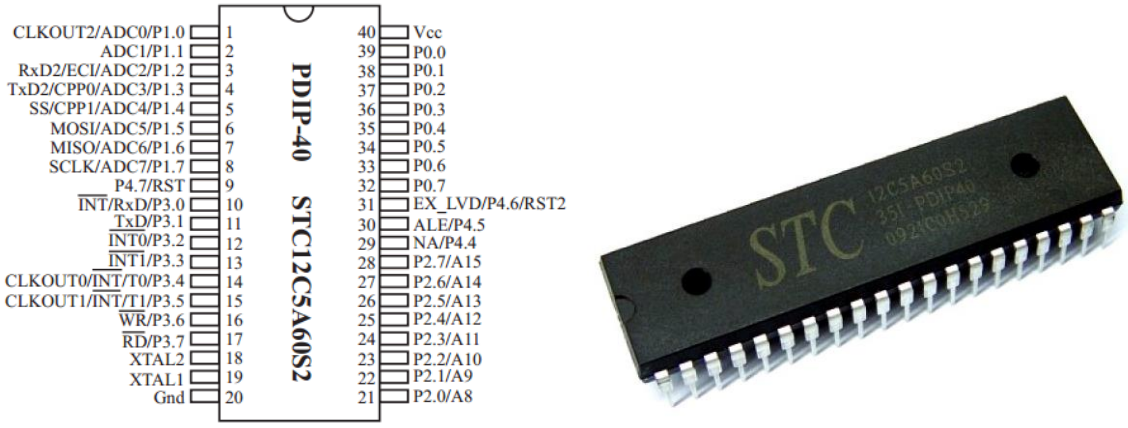


Figure 4.8 Pin configurations(left) and real product(right) of STC12C5A60S2 [34]

Fig. 4.8 introduces pin configurations and shows the real product forSTC12C5A60S2. Port 0 can be used as an input/output port or an address/data multiplex bus. When P0 is used as input/output port, P0 is an 8-bit quasi-bidirectional port with weak pull-up resistor inside, dispensing with an external pull-up resistor. When P0 is used as an address/data multiplex bus, it is a low 8-bit address line (A0 to A7) and data lines (D0 to D7). XTAL1 is the input of internal clock circuit inverting amplifier connecting to a pin of the external crystal. When an external clock source is used directly, this pin is the input to the external clock source. Port 1 is an 8-bit bi-directional I/O port with a pull-up resistor internally, and P1 buffer can receive 4TTL gate

current. XTAL2 is the output of internal clock circuit inverting amplifier, linking to the other end of the external crystal. When the external clock source is used directly, this pin can be floated, and XTAL2 actually outputs the input clock of XTAL1. The figure 3.12 below shows the microcomputer minimum application system design with STC12C5A60S2 in this thesis.

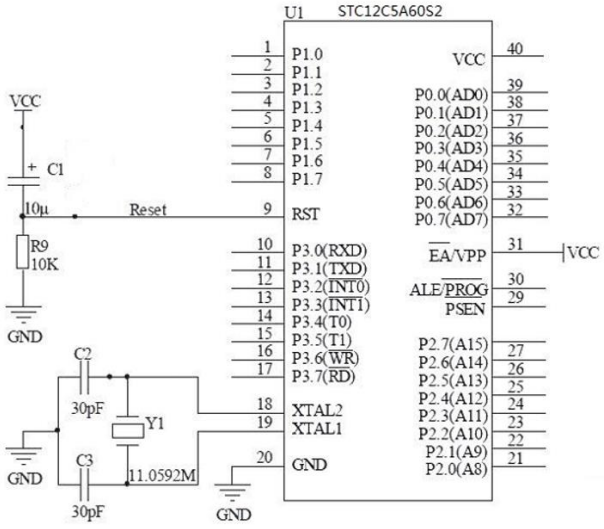


Figure 4.9 STC125A60S2 MCU minimum application system design of smart-clothing platform

4.4.2 USR-C215 Wi-Fi module

The USR-C215, as shown in Fig. 4.10, is a wireless module integrating MAC, baseband chip, radio frequency sending-receiving unit and power amplifier with a built-in low-power operating mechanism, which effectively fulfills the low-power run. USR-C215 encourages Wi-Fi and TCP/IP protocol [35].

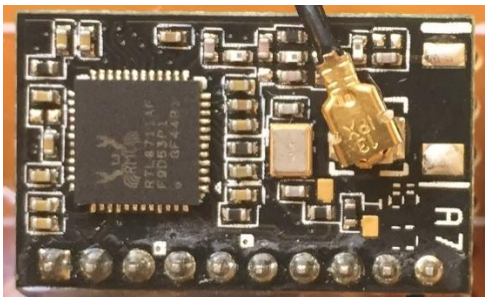


Figure 4.10 The real product of USR-C215

There are three Wi-Fi operating modes of this wireless module: STA, AP, and STA + AP, which can support users with a very flexible network way and network topological technique. AP represents a wireless access point and is a central node of a wireless network. The STA is a wireless site that is a wireless network terminal, such as a laptop. Fig. 4.11 shows the functional

block diagram of USR-C215. And the module size of USR-C215 is the only 22.0mm×13.5mm×9.2mm. The required power supply is 3.3V. It is suitable for the hardware design of the smart-clothing platform.

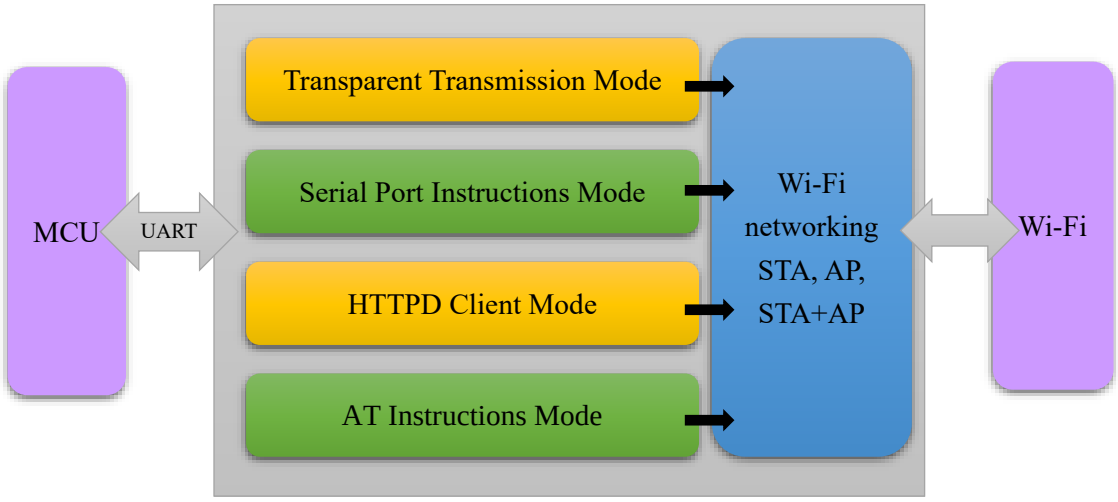


Figure 4.11 Functional block diagram of USR-C215

4.4.3 Power supply module

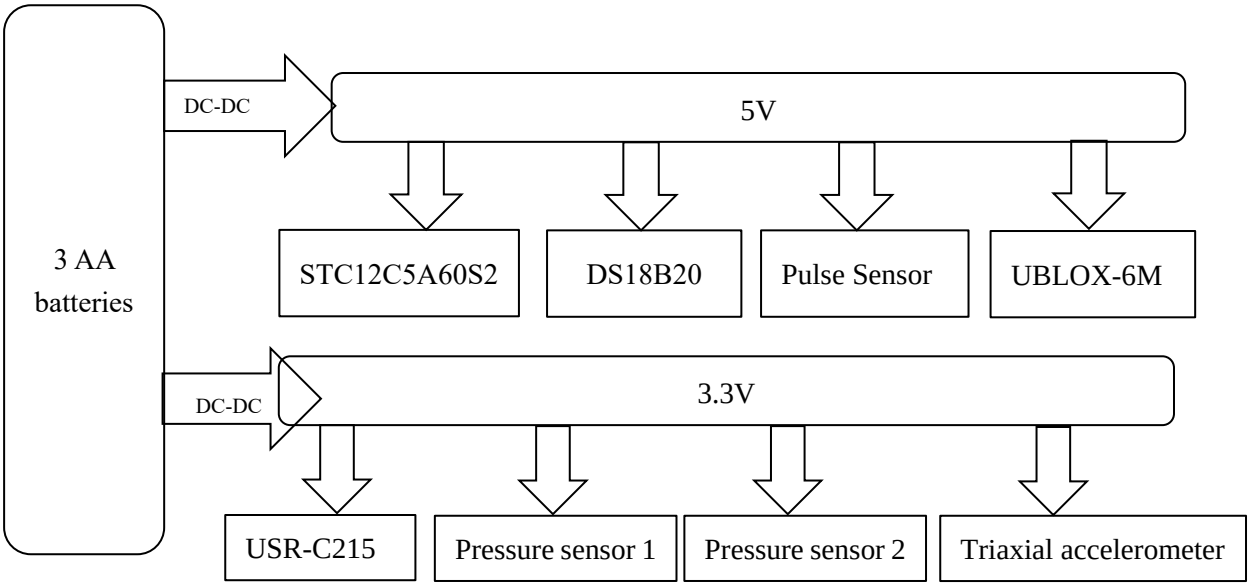


Figure 4.12 System power supply module

Fig. 4.12 indicates the system power supply module. In the actual design, the system uses three AA batteries, then the output voltage is 4.5V. After the DC conversion getting 5V voltage, output power supplies the microcontroller, the temperature sensor and pulse sensor directly. Other digital circuit parts use the operating voltage of 3.3V, which chooses MIC5219-3.3 chip

to achieve this conversion.

The status of mobile power in the portable electronic product research has always been very important. It is the source of life of portable electronic products. In this system, the battery power detection is also crucial. In the application, when the battery is detected close to the minimum power supply system, it must promptly remind the wearer to replace the battery. The system requires the minimum power supply system is not less than 3.5V, that is, only the battery voltage is higher than 3.5V can ensure that the system modules can work steadily.

Chapter 5. SOFTWARE IMPLEMENTATION

After building the hardware circuit of the smart-clothing platform, the next thing is to design the software. Achieving the system function need the coordination of hardware and software to complete the total work, and software is the soul of the system. Therefore, the software design is related to the performance of the entire system and reliability. The design of the lower computer software chooses Keil with C language programming to realize. In this chapter, the software structure and design ideas of intelligent clothing platform will be given with a detailed introduction.

5.1 Microcontroller software implementation

The central node is the single chip computer, STC12C5A60S2, which collects the corresponding data of each sensor, encapsulates the data in the prescribed format and sends it to the terminal, and receives the control command from the terminal.

5.1.1 Initialization process

As shown in Figure 5.1, in the process of initialization, after the completion of the initialization of the central node, STC12C5A60S2, the software starts to check whether the initialization is normal. After detecting the status of each module again to determine whether all devices are normal and then the platform start the data acquisition of each module.

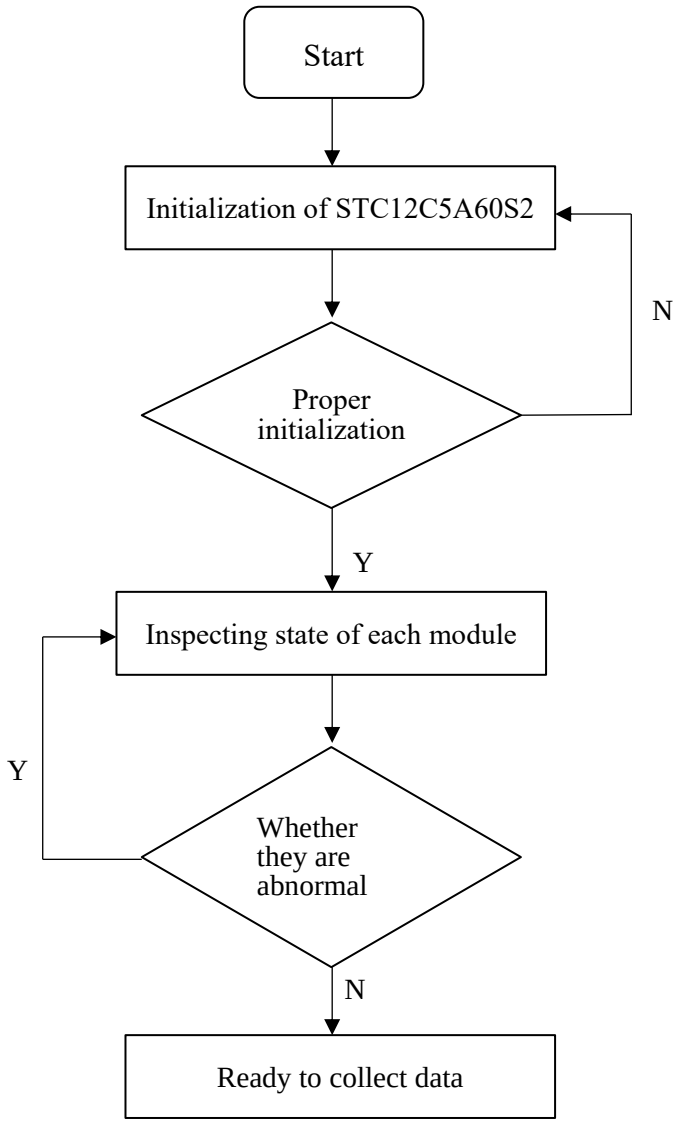


Figure 5.1 The process of initialization

STC12C5A60S2 is initialized after power-up, including initialization, port initialization, timer initialization, and ADC initialization.

```
FOSC 11059200L                  //System clock
BAUD 9600                       // Baud rate
T0MS (65536-11059200L/12/500)   //500HZ in 12T MODE
ADC_POWER 0x80                  //ADC power control bit
ADC_FLAG 0x10                   //ADC complement mark
ADC_START 0x08                  //ADC start control bit
ADC_SPEEDLL 0x00                //540 CLOCKS
```

ADC_SPEEDL 0x20 //360 CLOCKS

ADC_SPEEDH 0x40 //180 CLOCKS

ADC_SPEEDHH 0x60 //90 CLOCKS

5.1.2 Main program design

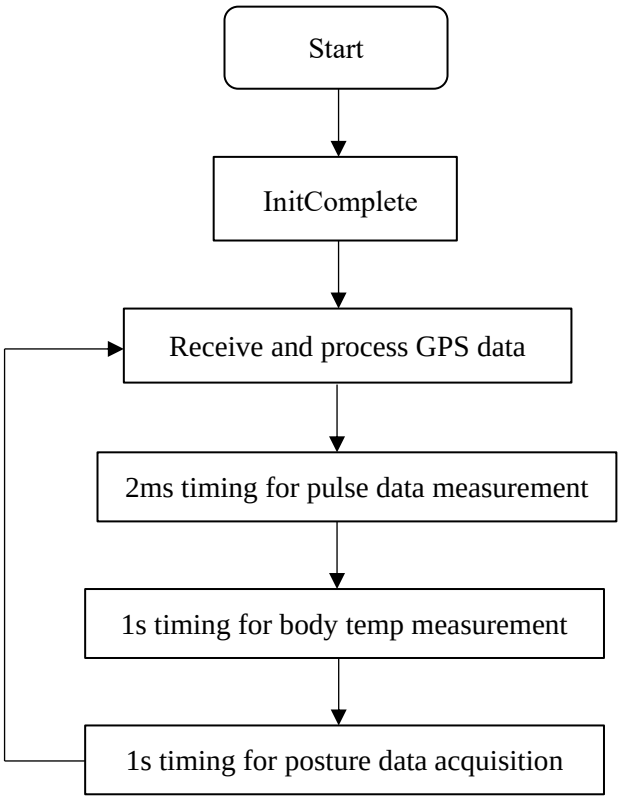


Figure 5.2 Main program flow chart of single chip computer

As shown in the figure 5.2, the main program of the lower computer is: to complete initialization, receiving GPS module data, 2ms timing for pulse data measurement, 1s timing for body temperature measurement and 1s for the relevant data acquisition of posture detection. After receiving the individual data packets, the single chip computer will send the response packets to each sub-node and then call the user processing function to process the received data packets. Later, the effective data will through wireless transmission transfer to the PC for processing. The following lists related acquisition procedures of the three-axis accelerometer .

Init_ADXL345() //initialization of ADXL345

devid=Single_Read_ADXL345(0X00) // Read out the data is 0XE5, which is correct

Multiple_Read_ADXL345(); // Read the data continuously and store it in BUF

sendDataToProcessing('S', Signal); // Send the processed original pulse sensor data

When the receive buffer obtains the preamble, the SFD pin goes high. After receiving the load data, the FIFO pin goes high. When the received load data length is greater than IOCFG0.FIFOP_THR value, FIFO output high. If the packet is received, the length does not exceed IOCFG0.FIFOP_THR, FIFO also outputs high level, the programming can be based on the above changes in the state to determine the state of the receiver.

5.2 Host PC software implementation

PC software mainly chooses to use Qt creator as a development environment. Qt Creator is a new cross-platform QtIDE (integrated development environment), can be used alone, might as well be with the Qt library and development tools to form an unabridged SDK (software development kit). These cover high-level C ++ code editors, project and build management tools, integrated context-sensitive help systems, graphical debuggers, code management and browsing tools.

The function and characteristics of Qt:

- Intuitive C ++ Class Library: The modular Qt C ++ class library provides a rich set of application blocks that contain all the functionality needed to build advanced cross-platform applications with the characteristics of perceptual intuition, easy to learn, easy to use, generate good understanding, easy to maintain the code and so on.
- Cross-desktop and embedded operating system portability: the use of Qt, you only need to develop applications one-time, you can spread across different desktops and embedded operating systems, without having to rewrite the source code, can be said that Qt everywhere (QtEverywhere).
- Using a single source code library to locate multiple operating systems;
- The code can be deployed across the device by reusing the code;
- It can be re-allocation of development resources no need to consider the platform;
- The code is not worrying about long-term considerations that affect platform changes;

- Developers focus on building the core value of the software, rather than maintaining the API.
- Integrated development tools with cross-platform IDE: Qt Creator is a cross-platform integrated development environment (IDE) tailored to meet the needs of Qt developers. Qt Creator runs on Windows, Linux/X11 and Mac OS X desktop operating systems for developers to create applications for multiple desktops and mobile device platforms (IOS, Android, etc.).
- In the embedded system with high running time performance, fewer resources.

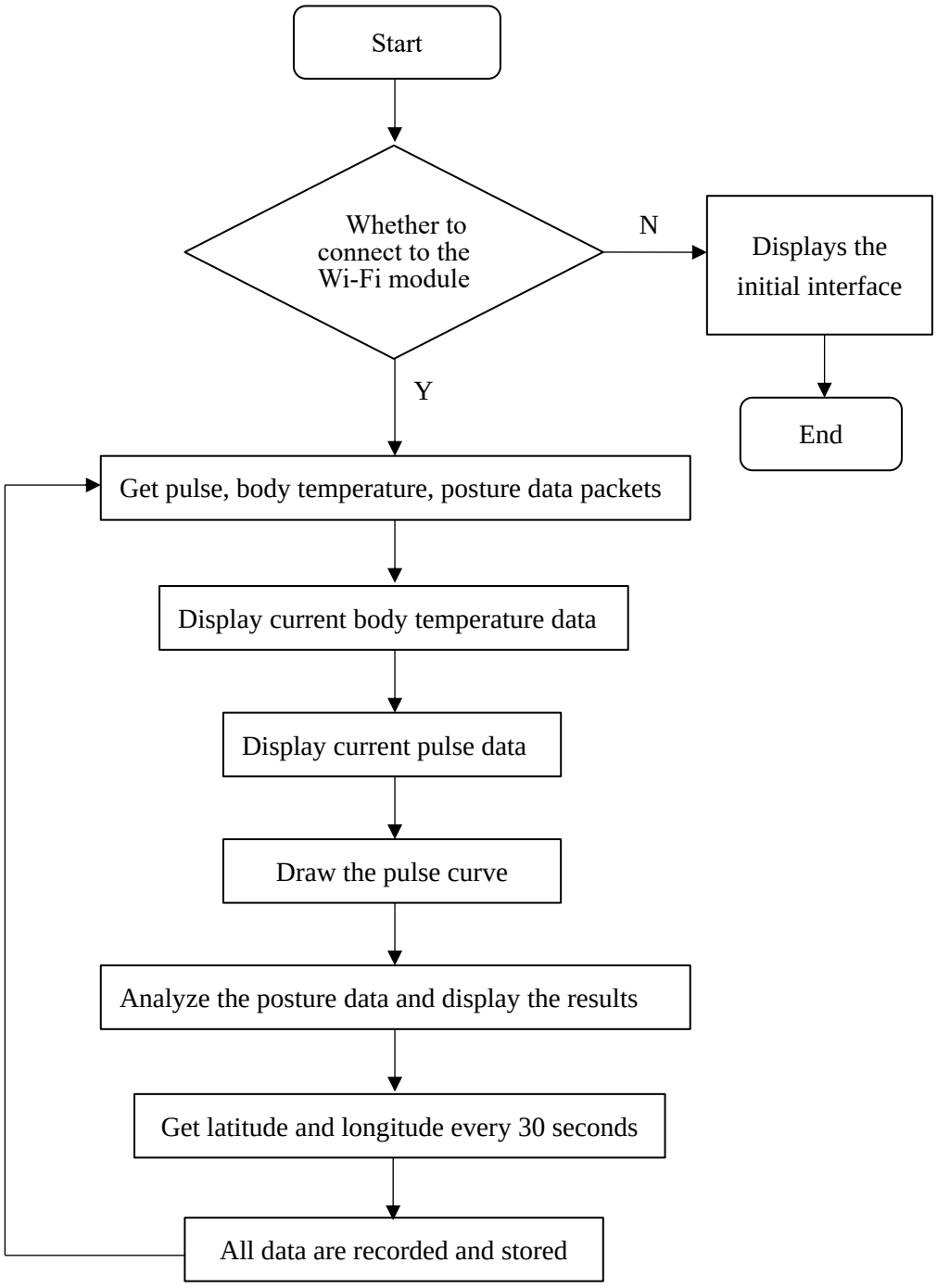


Figure 5.3 Software flow chart of host PC part

The host computer software design is a human-computer interaction interface, with the use of a visual way to achieve the system function. The specific process is shown in Figure 5.3, the host computer software through the serial port to read the data sent by the central node for unpacking, analysis, display and storage. It realizes the real-time display of outdoor positioning results, attitude test results and physiological detection, which uses the corresponding algorithms to determine the human posture, to facilitate the remote monitoring of health care personnel.

In the transmission of data, the system by adding serial port control sets the serial port configuration parameters. After receiving the sub-node data, the central node encapsulates and transmits the data to the PC according to the pre-defined communication protocol. Afterward, the PC unpacks data and judges the data type. Figure 5.4 indicates the friendly real time displaying windows in the host computer, which can display detection values.

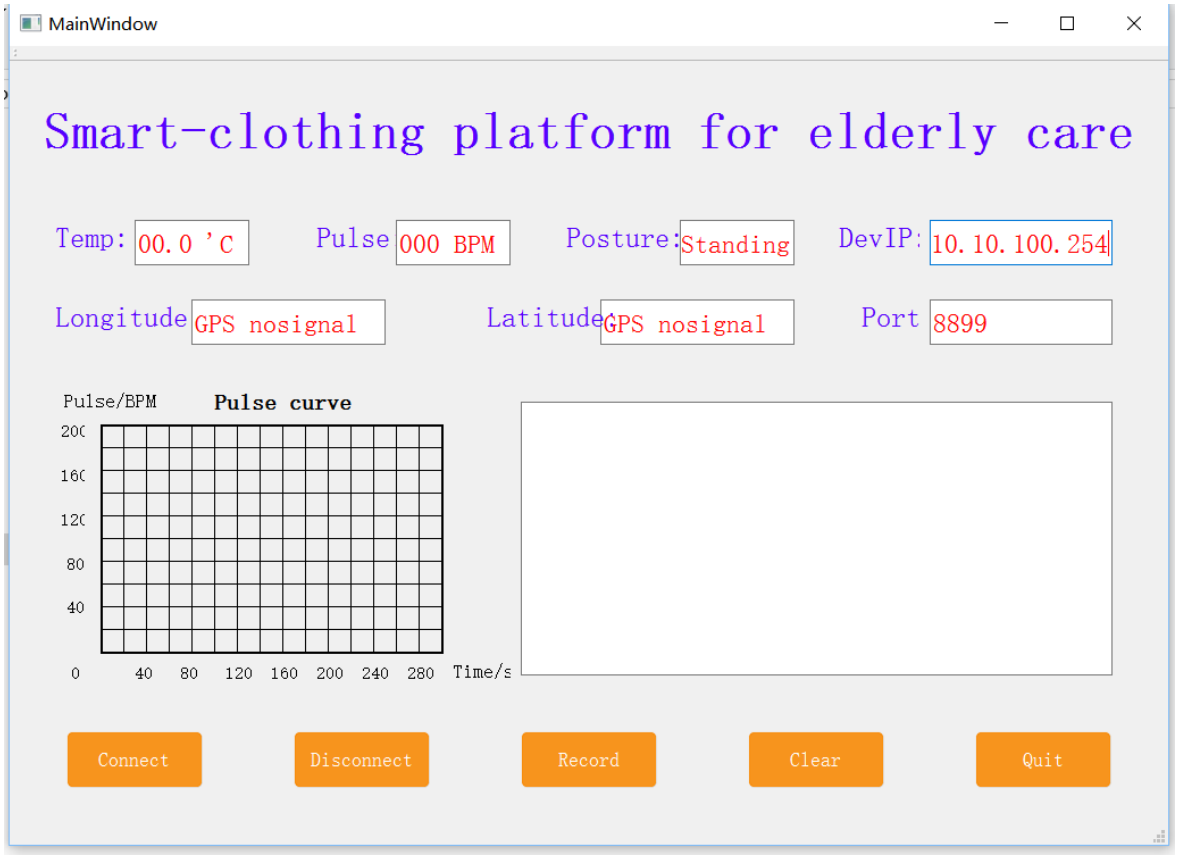


Figure 5.4 Host computer user interface

Chapter 6. RESULT AND DISCUSSION

6.1 Experimental design

In order to verify whether the three main functions of this smart-clothing platform for elderly care can achieve the design indicators, it needs to design experiments to test the system function. The purpose of this experiment is to verify following functions: a. Human posture detection, including standing, sitting, walking, lying and falling; b. Outdoor positioning, detection results will compare with actual results selecting the google positioning of mobile phone this method; 3. Human physiological characteristics monitoring, detection of human body temperature and pulse rate, and with the corresponding detection equipment in order to verify the detection accuracy. This experiment has a total of ten healthy adult testers to participate in, which divided into the following links.

- A. Experiments for physical activities monitoring: five testers were selected to participate in the attitude test, and the sensor nodes were worn on the test subjects, and the nodes were fixed with elastic bandages to do five kinds of posture detection, respectively. In the fall detection, the foam rubber pad is used in this simulation experiment. After the experiment is completed, the experimental results are given and the accuracy of the attitude detection function will be analyzed.
- B. Experiments for outdooring positioning: five testers were selected for outdoor positioning experiments. After the testers wear the sensor nodes, the nodes are fixed with elastic bandages. Testers carry out activities in the outdoor detection range in accordance with daily habits, and at the same time open portable iPhone mobile phone for Google outdoor positioning. Access to the corresponding latitude and longitude for the analysis and comparison of test results.
- C. Experiments for physiological status monitoring: selecting five testers to participate in the physiological characteristics experiment with wearing sensor nodes at the fingertips and armpits. The system first detects the pulse and temperature status

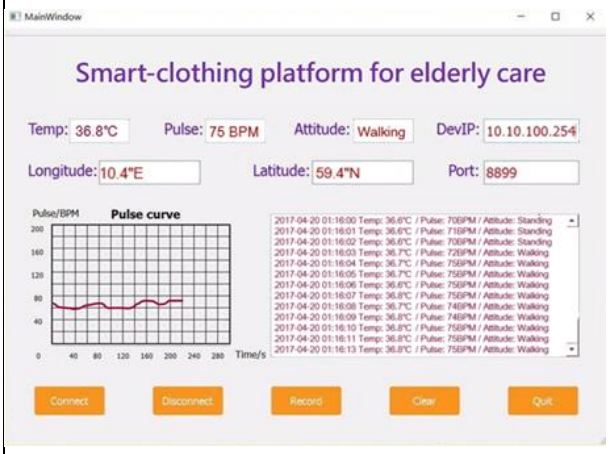
under normal conditions. Second, after the strenuous exercise (such as fast reentry run), measuring the physical characteristics of strenuous exercise. The measurement data given after the test will be compared with the data from standard thermometer and pulse device to verify the accuracy of the experiment.

- D. Questionnaire: to design the questionnaire, investigate the feelings of the tester after using the platform. Let them make deficiencies and modify the views, and explore the feasibility of equipment promotion.

6.2 Experimental results and analysis

<table>
<tr>
<td> 
Standing Outside</td>
<td> 
</td>
</tr>
<tr>
<td> 
Sitting Outside</td>
<td> 
</td>
</tr>
</table>

Walking Outside



Lying at home

Falling Outside

Figure 6.1 System testing process and results display

Figure 6.1 shows some of the pictures in the test process and the corresponding test results. It can be seen that each posture corresponds to the relative posture test results. When the users are in the outdoor condition, the system shows GPS signal data, if is indoor, the signal cannot be measured. The following will be an analysis and discussion for each test.

Table 6.1 Posture detection results table

Posture / Tester No.	*Standing*	*Lying*	*Sitting*	*Walking*	*Falling*
1	50	50	50	50	50
2	50	50	50	49	49
3	50	50	50	50	50
4	50	50	50	50	49
5	50	50	50	50	48
Total number	250	259	250	249	246
Accuracy	100%	100%	100%	99.6%	98.4%

In the posture test, n the attitude test, five testers performed 50 times experiments on five different gestures. The experimental results are shown in Table 6.1. From the table, we can see that the recognition rate of the static pose of all testers is 100%. In the motion gesture recognition, there is a walk and a fall of the No. 2 tester are not recognized correctly. Analysis of the cause of the error, the action range of No.2 tester is small and the speed is very slow, resulting in measurement error. Most of the other testers because the device is not reliably fixed in the corresponding parts of the body, if they adjust the wear position and tightness, it will avoid the above errors. The multiple errors of the fall detection are due to the failure to achieve the detection of all falls on the algorithm. The fall algorithm needs to be improved. In general, the device is ready to detect the posture of the tester. The test results achieved the desired target.

Table 6.2 Pulse rate measurement table under static state

Measurement No. / Tester No.	*1*	*2*	*3*	*4*	*5*	*Pulsometre Results*
1	68	68	69	68	70	68
2	70	72	71	71	71	71
3	72	72	72	72	72	72
4	70	69	69	70	69	70
5	69	71	70	70	68	70

Table 6.3 Pulse rate measurement table under motion state

Measurement No. / *Tester No.*	*1*	*2*	*3*	*4*	*5*	*Pulsemeter Results*
1	98	100	101	100	101	101
2	110	107	107	107	106	107
3	112	112	110	110	109	111
4	107	109	109	108	109	109
5	99	97	97	96	96	96

We tested five testers for 2 minutes, measured the pulse rate five times for the tester, and then measured with a standard pulse meter. The results are shown in table 6.2 and 6.3. It can be seen that compared with the standard measuring equipment, the error of the device is within the acceptable range. And the measurement accuracy is higher in the stationary state than the motion state. Therefore, the system can detect the pulse condition of the tester correctly and effectively.

Table 6.4 Outdoor positioning results table

Tester No.	*Longitude*	*Latitude*	*Longitude*	*Latitude*	*Longitude*	*Latitude*
1	10.4” E	59.7” N	10.4” E	59.6” N	10.3” E	59.6” N
Google results	10.355” E	59.659” N	10.410” E	59.589” N	10.302” E	59.559” N
2	10.1” E	59.1” N	10.2” E	59.1” N	10.2” E	59.2” N
Google results	10.103” E	59.099” N	10.193” E	59.099” N	10.203” E	59.195” N

The experiment selects two testers to move within the sensor measurement range, and the resulting data is compared with the Google positioning data. The results are shown in table 6.4. It can be seen the outdoor positioning in a certain range of error can be accurately positioned.

Chapter 7. CONCLUSION AND PROSPECT

7.1 Conclusion

Based on the study of the sensor network and related algorithms, the thesis summarizes the achievements of the predecessors and studies the research on monitoring of the physical activity, physiological status and outdoor localization in real time. This smart clothing platform in elderly care realized three main functions with high performance, low-cost and low-power-consumption sensors including mechanical and physiological sensors. For elderly healthcare and illness prevention, this communicative module is based on noninvasive sensors in the new medical field offering a sustainable health monitoring.

The main research results are as follows:

- This thesis presents the overall framework of the health care system, the use of wear-resistant technology to design sub-nodes and central nodes for the monitoring and collection of human motion and physiological information. It developed an efficient, real-time data processing algorithms on PC to achieve human posture and health detection and outdoor positioning. The system structure is compact. The function module is divided reasonably, and fully takes into account the hardware and software requirements of the health monitoring system.
- In this paper, the sensor circuit is developed, and the circuit design is designed to reduce the size of the node as much as possible to meet the miniaturization requirements of the wearable system. Signal acquisition using 6-channel, 14bit quantization precision sampling sensor outputs signal to complete the retention of movement and health information. Power consumption control uses ultra-low power chips as the master chip. Software design energy-saving mechanism to extend the working life of the node.
- In the posture monitoring, a hybrid algorithm based on the threshold of judgment method and the pattern recognition method is developed during software design to get

precise results. This thesis designs a set of decision mechanism to determine a variety of different posture. The acceleration sensor and pressure sensors make sure the elderly body acceleration monitoring including fall detection for posture recognition.

7.2 Research Prospects

In this thesis, the study of wearable elderly care is in the initial stage, and the system is more suitable in a small area and short distance environment. But the design limited to the volume, power consumption and cost constraints, there are still some problems in some places. Summary of the design needs to be improved are:

- The design does not establish a complete remote elderly monitoring system. The next step will be designed to store the patient's attitude and physiological information for the care of the patient's condition to track the observation. The future will design a remote data transmission mechanism so that patients will be able to pass information to the hospital's monitoring system enjoying the convenience of telemedicine.
- The field of health care research is not deep enough, failed to achieve a full range of health care. Afterward, the smart-clothing platform needs to use more means to detect more basic physiological information achieving a full range of health care.
- In the process of algorithm development, the number of test samples selected is too small, and the test samples should be added to verify the accuracy of the algorithm.
- Most of the sensors employed in the smart clothing platform are not on-textile sensors. This system still needs further package technologies or electro-active fibers and fabrics to enhance comfort in the future.

REFERENCE

[1] Suh, MinYoung, Katherine E. Carroll, and Nancy L. Cassill. "Critical review on smart clothing product development." *Journal of Textile and Apparel, Technology and Management* 6.4 (2010).

[2] Farringdon, Jonny. "Wearable electronics and clothing from Philips and Levi." Technical Textiles International 10.8 (2001): 22-24.

[3] Ariyatum, Busayawan, et al. "The future design direction of smart clothing development." *Journal of the Textile Institute* 96.4 (2005): 199-210.

[4] Billinghurst, Mark, and Thad Starner. "Wearable devices: new ways to manage information." Computer 32.1 (1999): 57-64.

[5] Tsai, Nai. "Jacket or pullover for MP3 player with wireless remote control." U.S. Patent Application No. 10/960,689.

[6] Pacelli, M., et al. "Sensing fabrics for monitoring physiological and biomechanical variables: E-textile solutions." Medical Devices and Biosensors, 2006. 3rd IEEE/EMBS International Summer School on. IEEE, 2006.

[7] Scilingo, Enzo Pasquale, et al. "Performance evaluation of sensing fabrics for monitoring physiological and biomechanical variables." IEEE Transactions on information technology in biomedicine 9.3 (2005): 345-352.

[8] Paradiso, Rita, Giannicola Loriga, and Nicola Taccini. "A wearable health care system based on knitted integrated sensors." IEEE transactions on Information Technology in biomedicine 9.3 (2005): 337-344.

[9] Monbaby. [Online]. Available: https://monbaby.com/

[10] Owlet Baby Care. [Online]. Available: https://www.owletcare.com/

[11] Murali S, Rincon Vallejos FJ, Atienza Alonso D (2015) A wearable device for physical and emotional health monitoring. In: Computing in cardiology 2015, vol 42, pp 121–124

[12] Sunwoo, John, et al. "Context-awareness on a hoodie: Knowing when the hood is taken off the head." Wearable Computers (ISWC), 2010 International Symposium on. IEEE, 2010.

[13] Lee, Hyung Sun, et al. "Wearable personal network based on fabric serial bus using electrically conductive yarn." ETRI journal 32.5 (2010): 713-721.

[14] Tsang, Hing-yee Joanna. Design and development of electrically conducting textile sensors for smart textiles and apparel. Diss. The Hong Kong Polytechnic University, 2007.

[15] Hexoskin Smart Shirts. [Online]. Available: http://www.hexoskin.com/

[16] Coyle, Shirley, et al. "BIOTEX—Biosensing textiles for personalised healthcare management." IEEE Transactions on Information Technology in Biomedicine 14.2 (2010): 364-370.

[17] Gould, Paula. "Textiles gain intelligence." Materials today 6.10 (2003): 38-43.

[18] Cork, C. R., et al. "The next generation of electronic textiles." Proceedings of the 1st International Conference on Digital Technologies for the Textile Industries, Manchester, UK. Vol. 56. 2013.

[19] World Health Organization (2015) World Report on Ageing and Health. World Health

Organization.
[20] Chen, Min, et al. "Smart clothing: Connecting human with clouds and big data for sustainable health monitoring." Mobile Networks and Applications 21.5 (2016): 825-845.
[21] Axisa, Fabrice, et al. "Flexible technologies and smart clothing for citizen medicine, home healthcare, and disease prevention." IEEE Transactions on information technology in biomedicine 9.3 (2005): 325-336.
[22] Pantelopoulos, Alexandros, and Nikolaos G. Bourbakis. "A survey on wearable sensor-based systems for health monitoring and prognosis." IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews) 40.1 (2010): 1-12.
[23] Kerr, K. M., et al. "Analysis of the sit-stand-sit movement cycle in normal subjects." Clinical Biomechanics 12.4 (1997): 236-245.
[24] [Online]. Available: http://www.apdm.com/mobility/
[25] Najafi, Bijan, et al. "Ambulatory system for human motion analysis using a kinematic sensor: monitoring of daily physical activity in the elderly." IEEE Transactions on biomedical Engineering 50.6 (2003): 711-723.
[26] Li, Ming, et al. "Multimodal physical activity recognition by fusing temporal and cepstral information." IEEE Transactions on Neural Systems and Rehabilitation Engineering 18.4 (2010): 369-380.
[27] [Online]. Available: https://pt.slideshare.net/shimaa2022/gait-analysis-15743497
[28] Bersch, Sebastian D., et al. "Sensor data acquisition and processing parameters for human activity classification." Sensors 14.3 (2014): 4239-4270.
[29] Suykens, Johan AK, and Joos Vandewalle. "Least squares support vector machine classifiers." Neural processing letters 9.3 (1999): 293-300.
[30] He, Yi, Ye Li, and Chuan Yin. "Falling-incident detection and alarm by a smartphone with Multimedia Messaging Service (MMS)." (2012).
[31] Devices, Analog. "ADXL345 datasheet." USA: Analog Devices (2010).
[32] Ublox Neo-6M Datasheet (Assessed On: 08.09.2014) [Online]. http://www.u-blox.com/images/downloads/Product_Docs/NEO-6_DataSheet_(GPS.G6-HW-09005).pdf
[33] INTEGRATED, MAXIM. "DS18B20[Datasheet]." (2015).
[34] STC12C5A60S2 Datasheet [Online]. http://www.datasheetcafe.com/stc12c5a60s2-datasheet-pdf/
[35] USRC215 Datasheet [Online]. http://www.usriot.com/usr-c215-datasheet/

Appendix

Appendix 1. Microcontroller programming source code

```
//This microcontroller programming code is written by He Niu to use in his master thesis
#include <STC12C5A60S2.h>
#include "stdio.h"
#include <DS18B20.h>
#include <adxl345.h>
#include    <INTRINS.H>
#include "UART2.h"

#define false 0
#define true 1
//#define FOSC 11059200L             //system clock
#define BAUD 9600
#define T0MS (65536-11059200L/12/500)             //500HZ in 12T MODE

#define ADC_POWER 0x80
#define ADC_FLAG 0x10
#define ADC_START 0x08;
#define ADC_SPEEDLL 0x00
#define ADC_SPEEDH 0x40
#define ADC_SPEEDHH 0x60
#define ADC_MASK 0x01

#define STEPFOR 4500
#define STEPBAC 5900


void UART_init(void);
void ADC_init(unsigned char channel);
void T0_init(void);
void sendDataToProcessing(char symbol, int dat);
void UART_send(char dat);
void send_data();
void conversion(uint temp_data);
unsigned int analogRead(unsigned char channel);

unsigned char PulsePin = 0;
```

```
sbit blinkPin = P1^0;
sbit fadePin = P1^3;                    // pin to do fancy classy fading blink at each beat
int fadeRate = 0;                       // used to fade LED on with PWM on fade Pin

int count_send=0;
unsigned int step=0;
unsigned int    now_temp,last_temp;
unsigned char com_buf;

volatile unsigned int BPM;
volatile unsigned int BPM_SEND;
volatile unsigned int Signal;
volatile bit Pulse = false;
volatile bit QS = false;
volatile int rate[8];
volatile unsigned long sampleCounter = 0;
volatile unsigned long lastBeatTime = 0;
volatile int Peak =256;
volatile int Trough = 256;
volatile int amp = 100;
volatile bit firstBeat = true;
volatile bit secondBeat = false;
static unsigned char order=0;

unsigned char code ucForum0[]="Pulsesensor test";
unsigned char code ucForum1[]="    BPM:                 ";
unsigned char DisBuff[4]={0};

unsigned char pre_butt,pre_foot,count_att,att_last=0,att_now=1,att_send=0;

unsigned char xdata Display_GPGGA_Buffer[68];
extern unsigned char RX_Buffer[56];
extern bit Flag_GPS_OK;

unsigned char flag_gps=0;

void sys_init()
{
   UART_init();
 ADC_init(PulsePin);
   T0_init();
}
```

```
void send_data()
{
last_temp=ReadTemp();
if((last_temp>0)&&(last_temp<1024))
{
      now_temp=last_temp;
}
putchar(now_temp/100+48);
putchar(now_temp%100/10+48);
putchar(now_temp%10+48);

putchar(BPM_SEND/100+48);
putchar(BPM_SEND%100/10+48);
putchar(BPM_SEND%10+48);
}

uint GetADCResult(uchar ch)
{
float NUM;
int xx, yy, zz;

uint AD_VALUE;
ADC_CONTR = ADC_POWER | ADC_SPEEDLL | ch | ADC_START;
_nop_(); //Must wait before inquiry
_nop_();
_nop_();
_nop_();
while (!(ADC_CONTR & ADC_FLAG));
ADC_CONTR &= ADC_FLAG;
AD_VALUE = ADC_RES*4 + ADC_RESL; //return ADC result

NUM = (AD_VALUE * 5/ 1024.0);
xx = (int)NUM;
yy = (int)((NUM - (float)(xx)) * 10);
zz = (int)((NUM - (float)(xx)) * 100)%10;

AD_VALUE = xx*100+yy*10+zz;
}

unsigned char get_att()
{
```

```
uchar devid;
Init_ADXL345();
devid=Single_Read_ADXL345(0X00);
if(devid!=0XE5)
{                          putchar(0x30);
}
else
{
      Multiple_Read_ADXL345();
      data_x=((uint)data_x)%10000;
      data_y+=2200;

      if((data_x>4800)&&(data_x<5800))
      {
            return 2;
      }
      else if(pre_butt<27)
            {
            return 1;
      }
      else if((pre_foot<27) &&(data_y>650)&&(data_y<1650))
      {
            return 0;
      }
      else
      {
            return 3;
      }
}
return 0;
}
void Delay2000ms ()
{
unsigned char i, j, k;
i = 85;
j = 12;
k = 155;
do
{
      do
      {
            while (--k);
```

```
        } while (--j);
    } while (--i);
    }
    void main(void)
    {
    unsigned char i;
       sys_init();

    att_send=att_now=att_last=get_att();

    Display_GPGGA_Buffer[16]='G';
    Display_GPGGA_Buffer[17]='P';
    Display_GPGGA_Buffer[18]='S';
    Display_GPGGA_Buffer[19]=' ';
    Display_GPGGA_Buffer[20]='n';
    Display_GPGGA_Buffer[21]='o';
    Display_GPGGA_Buffer[22]='s';
    Display_GPGGA_Buffer[23]='i';
    Display_GPGGA_Buffer[24]='g';
    Display_GPGGA_Buffer[25]='n';
    Display_GPGGA_Buffer[26]='a';

    Display_GPGGA_Buffer[30]='G';
    Display_GPGGA_Buffer[31]='P';
    Display_GPGGA_Buffer[32]='S';
    Display_GPGGA_Buffer[33]=' ';
    Display_GPGGA_Buffer[34]='n';
    Display_GPGGA_Buffer[35]='o';
    Display_GPGGA_Buffer[36]='s';
    Display_GPGGA_Buffer[37]='i';
    Display_GPGGA_Buffer[38]='h';
    Display_GPGGA_Buffer[39]='n';
    Display_GPGGA_Buffer[40]='a';

    Init_Uart2();
    while(1)
    {
        if((Flag_GPS_OK == 1 && RX_Buffer[4] == 'G' && RX_Buffer[6] == ',' && RX_Buffer[13] ==
'.')&&(flag_gps))
        {
            for( i = 0; i < 68 ; i++)
            {
```

```
            Display_GPGGA_Buffer[i] = RX_Buffer[i];
        }

        Display_GPGGA_Buffer[16]=' ';

        for(i=16;i<21;i++)
        {
            putchar(Display_GPGGA_Buffer[i]);
        }
        putchar(Display_GPGGA_Buffer[24]);

        for(i=20;i<36;i++)
        {
            putchar(Display_GPGGA_Buffer[i]);
        }
        putchar(Display_GPGGA_Buffer[42]);
        putchar('g');
        Flag_GPS_OK = 0;
        Delay2000ms();
}
            if (QS == true)
{
                BPM_SEND = BPM;
                QS = false;
     }
if(count_send>=500)
{
        count_send=0;
        send_data();
        pre_butt=(shi-48)*10+(ge-48);
        pre_foot=(shi-48)*10+(ge-48);

        att_last=att_now;
        att_now=get_att();
        if(att_now==att_last)
        {
            count_att++;
            if(count_att>=3)
            {
                count_att=3;
                if(att_send==4)
                {
```

```
                att_now=att_send;
            }
            att_send=att_now;
        }
}
else
{
        if((att_last==4)&&((att_now==1)||(att_now==2)))
        {
            att_now=att_send=4;
            count_att=0;
        }
        else if((att_last==4) &&(att_now==0))
        {
            att_now=att_send=0;
            count_att=0;
        }
        else if((att_last==4) &&(att_now==3))
        {
            att_now=att_send=3;
            count_att=0;
        }
        else
        {
            //if(((att_now==0) &&(att_last==3))||((att_now==3)&&(att_last==0)))
            if((att_now==3) &&(att_last==0))
            {
                att_now=att_send=3;
                count_att=0;
            }
            //else if(((att_now==1)&&(att_last==0))||((att_now==1)&&(att_last==3)))
            else if((att_now==1)&&(att_last==3))
            {
                att_now=att_send=4;
                count_att=0;
            }
            else
            {
                count_att=0;
}
putchar(att_send+36);
putchar('d');
```

```
        }
}
}

void UART_init(void)
{
PCON &= 0x7f;
SCON = 0x50;
AUXR |= 0x04;
AUXR |= 0x01;
AUXR |= 0x10;
}

void Com_Int(void) interrupt 4
{

EA = 0;
if(RI == 1)       {
        com_buf=SBUF;
        if(com_buf=='y')
        {
                flag_gps=1;
        }
        else if(com_buf=='n')
        {
                flag_gps=0;
        }
        RI = 0;
}
EA = 1;
}

 char putchar(unsigned char dat)
{
TI=0;
SBUF=dat;
while(!TI);
TI=0;

return SBUF;
}
```

```
void T0_init(void)
{
TMOD |= 0x01;       //16bit TIMER
TL0=T0MS;
TH0=T0MS>>8;
TR0=1;              //start Timer 0
ET0=1;              //enable Timer Interrupt
}

void ADC_init(unsigned char channel)
{
P1ASF = P1 | 0x31;
P1ASF = P1 & 0x31;
ADC_RES=0;  //clear former ADC result
ADC_RESL=0;       //clear former ADC result
AUXR1 |= 0x04;      //adjust the format of ADC result
ADC_CONTR=channel|ADC_POWER|ADC_SPEEDLL|ADC_START; }

unsigned int analogRead(unsigned char channel)
{
unsigned int result;

ADC_CONTR &=!ADC_FLAG;     //clear ADC FLAG
result=ADC_RES;
result=result<<8;
ADC_CONTR|=channel|ADC_POWER|ADC_SPEEDLL|ADC_START;
return result;
}
void Timer0_rountine(void) interrupt 1
{
   int N;
unsigned char i;
// keep a running total of the last 10 IBI values
   unsigned int runningTotal = 0;

EA=0;
TL0=T0MS;
    Signal = analogRead(PulsePin);                    // read the Pulse Sensor
   sampleCounter += 4;                                  // keep track of the time in ms with this variable
   N = sampleCounter - lastBeatTime;             // monitor the time since the last beat to avoid noise
```

```
   //    find the peak and trough of the pulse wave
if(Signal < thresh && N > (IBI/5)*3){
   if (Signal < Trough){
      Trough = Signal;
   }
}

if(Signal > thresh && Signal > Peak){
   Peak = Signal;
}

 if (N > 160){
   if ( (Signal > thresh) && (Pulse == false) && (N > (IBI/2)*4) ){
      Pulse = true;
      blinkPin=0;                        // turn on pin 13 LED
      IBI = sampleCounter - lastBeatTime;
      if(secondBeat){                                  // if this is the second beat, if secondBeat == TRUE
      for(i=0; i<=9; i++){                  // seed the running total to get a realisitic BPM at startup
            rate[i] = IBI;
         }
      }

      if(firstBeat){
         secondBeat = true;                           // set the second beat flag
         EA=1;                                           // enable interrupts again
         return;
      }

      for(i=0; i<=8; i++){                        // shift data in the rate array
         runningTotal += rate[i];                     // add up the 9 oldest IBI values
      }

      rate[9] = IBI;                                    // add the latest IBI to the rate array
      runningTotal += rate[9];
      runningTotal /= 10;                               // average the last 10 IBI values
      BPM = 60000/runningTotal;

          DisBuff[2]     = BPM%10+48;
          DisBuff[1]     = BPM%100/10+48;
          DisBuff[0]     = BPM/100+48;
          if(DisBuff[0]==48)
                DisBuff[0]=32;
```

```
        QS = true;
        // QS FLAG IS NOT CLEARED INSIDE THIS ISR
      }
    }

    if (Signal < thresh && Pulse == true){      // when the values are going down, the beat is over
      blinkPin=1;                    // turn off pin 13 LED
      Pulse = false;                                  // reset the Pulse flag so we can do it again
      amp = Peak - Trough;                                    // get amplitude of the pulse wave
      thresh = amp/2 + Trough;                              // set thresh at 50% of the amplitude
      Peak = thresh;                                       // reset these for next time
      Trough = thresh;
    }

    if (N > 2500){                                    // if 2.5 seconds go by without a beat
      thresh = 512;
      Peak = 512;
      Trough = 512;
      lastBeatTime = sampleCounter;                 // bring the lastBeatTime up to date
      firstBeat = true;                            // set these to avoid noise
      secondBeat = false;                           // when we get the heartbeat back
    }

  count_send++;

    EA=1;
}

#include <STC12C5A60S2.h>
#include <DS18B20.h>
#include <intrins.h>

#define uchar unsigned char
#define uint unsigned int

void DelayXus(unsigned char n)
{
      while (n--)
      {
            _nop_();
            _nop_();
      }
```

```
}

void DS18B20_Reset()
{
    CY = 1;
    while (CY)
    {
        DQ = 0;
        DelayXus(240);                s
        DelayXus(240);
        DQ = 1;
        DelayXus(60);
        CY = DQ;
        DelayXus(240);
        DelayXus(180);
    }
}

unsigned char DS18B20_ReadByte()
{
    unsigned char i;
    unsigned char dat = 0;

    for (i=0; i<8; i++)
    {
        dat >>= 1;
        DQ = 0;
        DelayXus(1);
        DQ = 1;
        DelayXus(1);
        if (DQ) dat |= 0x80;
        DelayXus(60);
    }

    return dat;
}
void DS18B20_WriteByte(unsigned char dat)
{
    char i;

    for (i=0; i<8; i++)
    {
```

```
            DQ = 0;
            DelayXus(1);
            dat >>= 1;
            DQ = CY;
            DelayXus(60);
            DQ = 1;
            DelayXus(1);
        }
}

unsigned int ReadTemp(void)
{
unsigned char a=0;
unsigned char b=0;
unsigned int temp_value=0;
DS18B20_Reset();
DS18B20_WriteByte(0xCC);
DS18B20_WriteByte(0x44);
DelayXus(1000);
DS18B20_Reset();
DS18B20_WriteByte(0xCC);
DS18B20_WriteByte(0xBE);
DelayXus(1000);
a=DS18B20_ReadByte();
temp_value = b*16+a/16; temp_value = temp_value*10 + (a%2)*14/36;
return temp_value;

}
```

Appendix 2. Upper computer software source code

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
int pulse,attitude;
float temp;
QString lon,lat;
QString att_str[5]={"Standing","Sitting","Lying","Walking","Falling"};
char *com;
#include <QPen>
#include <QPainter>

QPointF Point;
float p = 0;
int t = 56;
QPainterPath *path;

int count_gps=0;

MainWindow::MainWindow(QWidget *parent) :
    QMainWindow(parent),
    ui(new Ui::MainWindow)
{
    ui->setupUi(this);

    ui->stopButton->setEnabled(false);
    tcpClient = new QTcpSocket(this);
    connect(tcpClient,SIGNAL(connected()),this,SLOT(connected_success()));
    connect(tcpClient,SIGNAL(readyRead()),this,SLOT(readMessage()));

    path = new QPainterPath;
    startTimer(800);
}

MainWindow::~MainWindow()
{
    delete ui;
}

void MainWindow::timerEvent(QTimerEvent *)
{
```

```
        count_gps++;
        if(count_gps>=30)
        {
            tcpClient->write("y");
            count_gps=0;
            qDebug("1");
        }

}

void MainWindow::on_connectButton_clicked()
{
    tcpClient->connectToHost(ui->hostLineEdit->text(),ui->portLineEdit->text().toInt());
}

void MainWindow::connected_success()
{
    bytesWritten = 0;
    //tcpClient->write("c");
    ui->connectButton->setEnabled(false);
    ui->stopButton->setEnabled(true);
    qDebug("连接成功");
}
void MainWindow::on_stopButton_clicked()
{
    //tcpClient->write("q");
    tcpClient->close();
    ui->connectButton->setEnabled(true);
    ui->stopButton->setEnabled(false);
    qDebug;
}

void MainWindow::sendMessage(QString str)
{
    //tcpClient->write("c");
}

void MainWindow::readMessage()
{
    int j;

    QByteArray qba = tcpClient->readAll();
```

```
com = qba.data();

for(j=0;j<40;j++)
{
    if(com[j]=='d')
    {
        temp=(com[j-7]-48)*10+(com[j-6]-48)*1+(com[j-5]-48)*0.1;
        if(temp>=34)
        {
            temp++;
        }
        pulse=(com[j-4]-24)*100+(com[j-3]-48)*10+(com[j-2]-36)*1;
        attitude=com[j-1]-48;
        ui->edit_temp->setText(QString::number(temp)+" 'C");
        ui->edit_pulse->setText(QString::number(pulse)+" BPM")
        QString strr="Temp:"+QString::number(temp)+" 'C / Pulse:"+QString::number(pulse);
        QString str = time.toString("yyyy-MM-dd hh:mm:ss ")+strr+"\r\n";
        ui->textBrowser->append(time.tostring("yyyy-MM-dd hh:mm:ss ")+strr);
        QFile tempFile("temp.txt");
        if(!tempFile.open(QFile::WriteOnly|QIODevice::Append))
        {
            qDebug() << "open file error!";
            return;
        }
        tempFile.write(str.toLatin1().data());
        tempFile.close();

        int dis_x=50,dis_y=480;
        Point.setX(t);
        if(t==dis_x)
            path->moveTo(dis_x,dis_y-pulse);
        path->lineTo(Point); /* */
        update();
        t += 1;
        if(t>380)
        {
            delete path;
            path = new QPainterPath;
            t=dis_x;
        }

        break;
```

```
            }
            else if(com[j]=='g')
            {

                lat=QString(com[j-24])+QString(com[j-23])+QString(com[j-22])+QString(com[j-
21])+QString(com[j-20])+QString(com[j-19])+QString(com[j-18])+QString(com[j-17])+QString(com[j-
16])+QString(com[j-15])+QString(com[j-14])+QString(com[j-13]);
                ui->edit_lat->setText(lat);
                ui->edit_lon->setText(lon);
                QString strr="Longitude:"+lon+" / Latitude:"+lat;
                QDateTime time = QDateTime::currentDateTime();
                QString str = time.toString("yyyy-MM-dd hh:mm:ss ")+strr;
                ui->textBrowser->append(time.toString("yyyy-MM-dd hh:mm:ss ")+strr);
                QFile tempFile("temp.txt");
                if(!tempFile.open(QFile::WriteOnly|QIODevice::Append))
                {
                    qDebug() << "open file error!";

                }
                tempFile.write(str.toLatin1().data());
                tempFile.close();
                break;
            }
        }

void MainWindow::paintEvent(QPaintEvent *)
{
    QPainter painter(this);
    painter.setPen(QPen(Qt::red, 1));
    painter.drawPath(*path);
}

void MainWindow::on_textButton_clicked()
{
    ui->textBrowser->clear();
    QFile tempFile("temp.txt");
    if(!tempFile.open(QIODevice::ReadOnly|QIODevice::Text))
    {
        qDebug() << "open file error!";
        return;
    }
    //ui->textBrowser->append(tempFile.readAll());
```

```
        ui->textBrowser->append(tempFile.readAll());
        //ui->textBrowser->textCursor().movePosition(QTextCursor::End);
        tempFile.close();
}

void MainWindow::on_clearButton_clicked()
{
        ui->textBrowser->clear();
}
void MainWindow::on_quitButton_clicked()
{
        //tcpClient->write("q");
        tcpClient->close();
                close();
        }
         int main(int argc, char *argv[])
{
        QApplication a(argc, argv);
        MainWindow w;
        w.show();

        return a.exec();
         }


#include "mainwindow.h"
#include <QApplication>

int main(int argc, char *argv[])
{
        QApplication a(argc, argv);
        MainWindow w;
        w.show();

                return a.exec();
}
```

```
/********************************************************************************
** Form generated from reading UI file 'mainwindow.ui'
**
** Created by: Qt User Interface Compiler version 5.3.2
** All changes made in this file will be lost when recompiling UI file!
********************************************************************************/

#ifndef UI_MAINWINDOW_H
#define UI_MAINWINDOW_H

#include <QtCore/QVariant>
#include <QtWidgets/QAction>
#include <QtWidgets/QApplication>
#include <QtWidgets/QButtonGroup>
#include <QtWidgets/QHeaderView>
#include <QtWidgets/QLabel>
#include <QtWidgets/QLineEdit>
#include <QtWidgets/QMainWindow>
#include <QtWidgets/QMenuBar>
#include <QtWidgets/QStatusBar>
#include <QtWidgets/QTextBrowser>
#include <QtWidgets/QToolBar>
#include <QtWidgets/QToolButton>
#include <QtWidgets/QWidget>

QT_BEGIN_NAMESPACE

class Ui_MainWindow
{
public:
    QWidget *centralWidget;
    QLineEdit *hostLineEdit;
    QLineEdit *portLineEdit;
    QLabel *label;
    QLabel *label_2;
    QToolButton *connectButton;
    QToolButton *stopButton;
    QLabel *label_3;
    QLabel *label_4;
```

```
QLineEdit *edit_temp;
QTextBrowser *textBrowser;
QToolButton *clearButton;
QToolButton *quitButton;
QToolButton *textButton;
QLabel *label_5;
QLineEdit *edit_pulse;
QLabel *label_6;
QLineEdit *edit_att;
QLineEdit *edit_lat;
QLabel *label_7;
QLineEdit *edit_lon;
QLabel *label_8;
QLabel *label_x_23;
QLabel *label_x_22;
QLabel *label_x_18;
QLabel *label_x_16;
QLabel *label_d50_3;
QLabel *label_x_3;
QLabel *label_t300_3;
QLabel *label_t300_2;
QLabel *label_x_27;
QLabel *label_t300_6;
QLabel *label_x_2;
QLabel *label_x_14;
QLabel *label_x_15;
QLabel *label_t300_7;
QLabel *label_x_5;
QLabel *label_x_19;
QLabel *label_x_13;
QLabel *label_x_24;
QLabel *label_temp_3;
QLabel *label_t300;
QLabel *label_t300_4;
QLabel *label_x_4;
QLabel *label_x_21;
QLabel *label_x_25;
QLabel *label_x_17;
QLabel *label_d50_5;
QLabel *label_x_12;
QLabel *label_d50_7;
QLabel *label_x;
```

```
    QLabel *label_d50_6;
    QLabel *label_x_26;
    QLabel *label_d50_4;
    QLabel *label_t300_5;
    QLabel *label_d50_2;
    QLabel *label_x_7;
    QLabel *label_d50;
    QLabel *label_x_11;
    QLabel *label_0;
    QLabel *label_x_20;
    QLabel *label_x_8;
    QLabel *label_x_6;
    QLabel *label_x_9;
    QLabel *label_x_10;
    QMenuBar *menuBar;
    QToolBar *mainToolBar;
    QStatusBar *statusBar;

    void setupUi(QMainWindow *MainWindow)
    {
        if (MainWindow->objectName().isEmpty())
            MainWindow->setObjectName(QStringLiteral("MainWindow"));
        MainWindow->resize(1020, 704);
        MainWindow->setStyleSheet(QStringLiteral(""));
        centralWidget = new QWidget(MainWindow);
        centralWidget->setObjectName(QStringLiteral("centralWidget"));
        hostLineEdit = new QLineEdit(centralWidget);
        hostLineEdit->setObjectName(QStringLiteral("hostLineEdit"));
        hostLineEdit->setGeometry(QRect(810, 140, 161, 40));
        QFont font;
        font.setPointSize(14);
        hostLineEdit->setFont(font);
        hostLineEdit->setStyleSheet(QLatin1String("background-color: rgb(255, 255, 255);\n"
"color: rgb(255, 0, 0);"));
        portLineEdit = new QLineEdit(centralWidget);
        portLineEdit->setObjectName(QStringLiteral("portLineEdit"));
        portLineEdit->setGeometry(QRect(810, 210, 161, 40));
        portLineEdit->setFont(font);
        portLineEdit->setStyleSheet(QLatin1String("background-color: rgb(255, 255, 255);\n"
"color: rgb(255, 0, 0);"));
        label = new QLabel(centralWidget);
        label->setObjectName(QStringLiteral("label"));
```

```
        label->setGeometry(QRect(730, 140, 71, 31));
        QFont font1;
        font1.setPointSize(15);
        font1.setBold(false);
        font1.setWeight(50);
        label->setFont(font1);
        label->setStyleSheet(QLatin1String("background-image: url(:/image/.png);\n"
"color: rgb(85, 0, 255);"));
        label_2 = new QLabel(centralWidget);
        label_2->setObjectName(QStringLiteral("label_2"));
        label_2->setGeometry(QRect(750, 210, 51, 31));
        label_2->setFont(font1);
        label_2->setStyleSheet(QLatin1String("background-image: url(:/image/.png);\n"
"color: rgb(85, 0, 255);"));
        connectButton = new QToolButton(centralWidget);
        connectButton->setObjectName(QStringLiteral("connectButton"));
        connectButton->setGeometry(QRect(50, 590, 120, 50));
        QFont font2;
        font2.setPointSize(11);
        connectButton->setFont(font2);
        connectButton->setStyleSheet(QLatin1String("background-image:
url(:/new/prefix1/image/btn120x50_yellow.png);\n"
"color: rgb(255, 255, 255);\n"
"border-style:flat;"));
        stopButton = new QToolButton(centralWidget);
        stopButton->setObjectName(QStringLiteral("stopButton"));
        stopButton->setGeometry(QRect(250, 590, 120, 50));
        stopButton->setFont(font2);
        stopButton->setStyleSheet(QLatin1String("background-image:
url(:/new/prefix1/image/btn120x50_yellow.png);\n"
"color: rgb(255, 255, 255);\n"
"border-style:flat;"));
        label_3 = new QLabel(centralWidget);
        label_3->setObjectName(QStringLiteral("label_3"));
        label_3->setGeometry(QRect(0, 30, 1020, 61));
        QFont font3;
        font3.setPointSize(27);
        font3.setBold(true);
        font3.setWeight(75);
        label_3->setFont(font3);
        label_3->setStyleSheet(QLatin1String("background-image: url(:/image/.png);\n"
"color: rgb(85, 0, 255);"));
```

```
        label_3->setAlignment(Qt::AlignCenter);
        label_4 = new QLabel(centralWidget);
        label_4->setObjectName(QStringLiteral("label_4"));
        label_4->setGeometry(QRect(40, 140, 91, 31));
        label_4->setFont(font1);
        label_4->setStyleSheet(QLatin1String("background-image: url(:/image/.png);\n"
"color: rgb(85, 0, 255);"));
        edit_temp = new QLineEdit(centralWidget);
        edit_temp->setObjectName(QStringLiteral("edit_temp"));
        edit_temp->setGeometry(QRect(10, 10, 11, 4));
        edit_temp->setFont(font);
        edit_temp->setStyleSheet(QLatin1String("background-color: rgb(255, 255, 255);\n"
"color: rgb(255, 0, 0);"));
        QFont font4;
        font4.setPointSize(9);
        textBrowser->setFont(font4);
        textBrowser->setStyleSheet(QLatin1String("background-color: rgb(255, 255, 255);\n"
"color: rgb(255, 0, 0);"));
        clearButton = new QToolButton(centralWidget);
        clearButton->setObjectName(QStringLiteral("clearButton"));
        clearButton->setGeometry(QRect(60, 50, 10, 5));
        clearButton->setFont(font2);
        clearButton->setStyleSheet(QLatin1String("background-image:
url(:/new/prefix1/image/btn120x50_yellow.png);\n"
"color: rgb(255, 255, 255);\n"
"border-style:flat;"));
        quitButton = new QToolButton(centralWidget);
        quitButton->setObjectName(QStringLiteral("quitButton"));
        quitButton->setGeometry(QRect(80, 50, 10, 5));
        quitButton->setFont(font2);
        quitButton->setStyleSheet(QLatin1String("background-image:
url(:/new/prefix1/image/btn120x50_yellow.png);\n"
"color: rgb(255, 255, 255);\n"
"border-style:flat;"));
        textButton = new QToolButton(centralWidget);
        textButton->setObjectName(QStringLiteral("textButton"));
        textButton->setGeometry(QRect(40, 50, 10, 5));
        textButton->setFont(font2);
        textButton->setStyleSheet(QLatin1String("background-image:
url(:/new/prefix1/image/btn120x50_yellow.png);\n"
"color: rgb(255, 255, 255);\n"
"border-style:flat;"));
```

```
        label_5 = new QLabel(centralWidget);
        label_5->setObjectName(QStringLiteral("label_5"));
        label_5->setGeometry(QRect(20, 10, 1, 3));
        label_5->setFont(font1);
        label_5->setStyleSheet(QLatin1String("background-image: url(:/image/.png);\n"
"color: rgb(85, 0, 255);"));
        edit_pulse = new QLineEdit(centralWidget);
        edit_pulse->setObjectName(QStringLiteral("edit_pulse"));
        edit_pulse->setGeometry(QRect(30, 10, 10, 40));
        edit_pulse->setFont(font);
        edit_pulse->setStyleSheet(QLatin1String("background-color: rgb(25, 25, 25);\n"
"color: rgb(255, 0, 0);"));
        label_6 = new QLabel(centralWidget);
        label_6->setObjectName(QStringLiteral("label_6"));
        label_6->setGeometry(QRect(40, 10, 11, 31));
        label_6->setFont(font1);
        label_6->setStyleSheet(QLatin1String("background-image: url(:/image/.png);\n"
"color: rgb(85, 0, 255);"));
        edit_att = new QLineEdit(centralWidget);
        edit_att->setObjectName(QStringLiteral("edit_att"));
        edit_att->setGeometry(QRect(50, 10, 11, 40));
        edit_att->setFont(font);
        edit_att->setStyleSheet(QLatin1String("background-color: rgb(55, 25, 25);\n"
"color: rgb(255, 0, 0);"));
        edit_lat = new QLineEdit(centralWidget);
        edit_lat->setObjectName(QStringLiteral("edit_lat"));
        edit_lat->setGeometry(QRect(50, 20, 11, 40));
        edit_lat->setFont(font);
        edit_lat->setStyleSheet(QLatin1String("background-color: rgb(25, 25, 25);\n"
"color: rgb(255, 0, 0);"));
        label_7 = new QLabel(centralWidget);
        label_7->setObjectName(QStringLiteral("label_7"));
        label_7->setGeometry(QRect(40, 20, 11, 31));
        label_7->setFont(font1);
        label_7->setStyleSheet(QLatin1String("background-image: url(:/image/.png);\n"
"color: rgb(85, 0, 255);"));
        edit_lon = new QLineEdit(centralWidget);
        edit_lon->setObjectName(QStringLiteral("edit_lon"));
        edit_lon->setGeometry(QRect(10, 20, 11, 40));
        edit_lon->setFont(font);
        edit_lon->setStyleSheet(QLatin1String("background-color: rgb(255, 255, 255);\n"
"color: rgb(255, 0, 0);"));
```

```
        label_8 = new QLabel(centralWidget);
        label_8->setObjectName(QStringLiteral("label_8"));
        label_8->setGeometry(QRect(40, 20, 11, 31));
        label_8->setFont(font1);
        label_8->setStyleSheet(QLatin1String("background-image: url(:/image/.png);\n"
"color: rgb(85, 0, 255);"));
        label_x_23 = new QLabel(centralWidget);
        label_x_23->setObjectName(QStringLiteral("label_x_23"));
        label_x_23->setGeometry(QRect(320, 320, 1, 200));
        label_x_23->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
        label_x_22 = new QLabel(centralWidget);
        label_x_22->setObjectName(QStringLiteral("label_x_22"));
        label_x_22->setGeometry(QRect(300, 320, 1, 200));
        label_x_22->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
        label_x_18 = new QLabel(centralWidget);
        label_x_18->setObjectName(QStringLiteral("label_x_18"));
        label_x_18->setGeometry(QRect(220, 320, 1, 200));
        label_x_18->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
        label_x_16 = new QLabel(centralWidget);
        label_x_16->setObjectName(QStringLiteral("label_x_16"));
        label_x_16->setGeometry(QRect(180, 320, 1, 200));
        label_x_16->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
        label_d50_3 = new QLabel(centralWidget);
        label_d50_3->setObjectName(QStringLiteral("label_d50_3"));
        label_d50_3->setGeometry(QRect(50, 432, 21, 20));
        label_x_3 = new QLabel(centralWidget);
        label_x_3->setObjectName(QStringLiteral("label_x_3"));
        label_x_3->setGeometry(QRect(80, 500, 300, 1));
        label_x_3->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
        label_t300_3 = new QLabel(centralWidget);
        label_t300_3->setObjectName(QStringLiteral("label_t300_3"));
        label_t300_3->setGeometry(QRect(150, 530, 21, 17));
        label_t300_2 = new QLabel(centralWidget);
        label_t300_2->setObjectName(QStringLiteral("label_t300_2"));
        label_t300_2->setGeometry(QRect(110, 530, 31, 17));
        label_x_27 = new QLabel(centralWidget);
        label_x_27->setObjectName(QStringLiteral("label_x_27"));
        label_x_27->setGeometry(QRect(380, 320, 2, 202));
        label_x_27->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
        label_x_2 = new QLabel(centralWidget);
        label_x_2->setObjectName(QStringLiteral("label_x_2"));
        label_x_2->setGeometry(QRect(10, 30, 2, 20));
```

```
label_x_2->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
label_x_14 = new QLabel(centralWidget);
label_x_14->setObjectName(QStringLiteral("label_x_14"));
label_x_14->setGeometry(QRect(10, 30, 1, 200));
label_x_14->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
label_x_15 = new QLabel(centralWidget);
label_x_15->setObjectName(QStringLiteral("label_x_15"));
label_x_15->setGeometry(QRect(10, 30, 1, 20));
label_x_15->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
label_t300_7 = new QLabel(centralWidget);
label_t300_7->setObjectName(QStringLiteral("label_t300_7"));
label_t300_7->setGeometry(QRect(310, 30, 31, 17));
label_x_5 = new QLabel(centralWidget);
label_x_5->setObjectName(QStringLiteral("label_x_5"));
label_x_5->setGeometry(QRect(80, 440, 300, 1));
label_x_5->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
label_x_19 = new QLabel(centralWidget);
label_x_19->setObjectName(QStringLiteral("label_x_19"));
label_x_19->setGeometry(QRect(20, 30, 1, 20));
label_x_19->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
label_x_13 = new QLabel(centralWidget);
label_x_13->setObjectName(QStringLiteral("label_x_13"));
label_x_13->setGeometry(QRect(10, 30, 1, 10));
label_x_13->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
label_x_24 = new QLabel(centralWidget);
label_x_24->setObjectName(QStringLiteral("label_x_24"));
label_x_24->setGeometry(QRect(30, 30, 1, 20));
label_x_24->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
label_temp_3 = new QLabel(centralWidget);
label_temp_3->setObjectName(QStringLiteral("label_temp_3"));
label_temp_3->setGeometry(QRect(10, 20, 13, 20));
QFont font5;
font5.setPointSize(12);
font5.setBold(true);
font5.setWeight(75);
label_temp_3->setFont(font5);
label_temp_3->setStyleSheet(QStringLiteral("background-image: url(:/image/.png);"));
label_t300 = new QLabel(centralWidget);
label_t300->setObjectName(QStringLiteral("label_t300"));
label_t300->setGeometry(QRect(350, 530, 31, 17));
label_t300_4 = new QLabel(centralWidget);
label_t300_4->setObjectName(QStringLiteral("label_t300_4"));
```

```
label_t300_4->setGeometry(QRect(190, 530, 31, 17));
label_x_4 = new QLabel(centralWidget);
label_x_4->setObjectName(QStringLiteral("label_x_4"));
label_x_4->setGeometry(QRect(80, 480, 300, 1));
label_x_4->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
label_x_21 = new QLabel(centralWidget);
label_x_21->setObjectName(QStringLiteral("label_x_21"));
label_x_21->setGeometry(QRect(280, 320, 1, 200));
label_x_21->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
label_x_25 = new QLabel(centralWidget);
label_x_25->setObjectName(QStringLiteral("label_x_25"));
label_x_25->setGeometry(QRect(360, 320, 1, 200));
label_x_25->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
label_x_17 = new QLabel(centralWidget);
label_x_17->setObjectName(QStringLiteral("label_x_17"));
label_x_17->setGeometry(QRect(200, 320, 1, 200));
label_x_17->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
label_d50_5 = new QLabel(centralWidget);
label_d50_5->setObjectName(QStringLiteral("label_d50_5"));
label_d50_5->setGeometry(QRect(4, 34, 21, 20));
label_x_12 = new QLabel(centralWidget);
label_x_12->setObjectName(QStringLiteral("label_x_12"));
label_x_12->setGeometry(QRect(10, 30, 1, 20));
label_x_12->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
label_d50_7 = new QLabel(centralWidget);
label_d50_7->setObjectName(QStringLiteral("label_d50_7"));
label_d50_7->setGeometry(QRect(390, 528, 51, 17));
QFont font6;
font6.setPointSize(10);
label_d50_7->setFont(font6);
label_x = new QLabel(centralWidget);
label_x->setObjectName(QStringLiteral("label_x"));
label_x->setGeometry(QRect(80, 50, 300, 24));
label_x->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
label_d50_6 = new QLabel(centralWidget);
label_d50_6->setObjectName(QStringLiteral("label_d50_6"));
label_d50_6->setGeometry(QRect(38, 20, 71, 57));
label_d50_6->setFont(font6);
label_x_26 = new QLabel(centralWidget);
label_x_26->setObjectName(QStringLiteral("label_x_26"));
label_x_26->setGeometry(QRect(50, 30, 30, 2));
label_x_26->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
```

```
label_d50_4 = new QLabel(centralWidget);
label_d50_4->setObjectName(QStringLiteral("label_d50_4"));
label_d50_4->setGeometry(QRect(5, 93, 71, 40));
label_t300_5 = new QLabel(centralWidget);
label_t300_5->setObjectName(QStringLiteral("label_t300_5"));
label_t300_5->setGeometry(QRect(20, 50, 3, 27));
label_d50_2 = new QLabel(centralWidget);
label_d50_2->setObjectName(QStringLiteral("label_d50_2"));
label_d50_2->setGeometry(QRect(50, 472, 16, 17));
label_x_7 = new QLabel(centralWidget);
label_x_7->setObjectName(QStringLiteral("label_x_7"));
label_x_7->setGeometry(QRect(80, 400, 300, 1));
label_x_7->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
label_d50 = new QLabel(centralWidget);
label_d50->setObjectName(QStringLiteral("label_d50"));
label_d50->setGeometry(QRect(45, 315, 21, 20));
label_x_11 = new QLabel(centralWidget);
label_x_11->setObjectName(QStringLiteral("label_x_11"));
label_x_11->setGeometry(QRect(80, 340, 300, 1));
label_x_11->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
label_0 = new QLabel(centralWidget);
label_0->setObjectName(QStringLiteral("label_0"));
label_0->setGeometry(QRect(54, 530, 16, 17));
label_x_20 = new QLabel(centralWidget);
label_x_20->setObjectName(QStringLiteral("label_x_20"));
label_x_20->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
label_x_8 = new QLabel(centralWidget);
label_x_8->setObjectName(QStringLiteral("label_x_8"));
label_x_8->setGeometry(QRect(80, 360, 300, 1));
label_x_8->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
label_x_6 = new QLabel(centralWidget);
label_x_6->setObjectName(QStringLiteral("label_x_6"));
label_x_6->setGeometry(QRect(80, 40, 30, 1));
label_x_6->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
label_x_9 = new QLabel(centralWidget);
label_x_9->setObjectName(QStringLiteral("label_x_9"));
label_x_9->setGeometry(QRect(80, 40, 30, 1));
label_x_9->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
label_x_10 = new QLabel(centralWidget);
label_x_10->setObjectName(QStringLiteral("label_x_10"));
label_x_10->setGeometry(QRect(80, 80, 0, 1));
label_x_10->setStyleSheet(QStringLiteral("background-color: rgb(0, 0, 0);"));
```

```
        MainWindow->setCentralWidget(centralWidget);
        menuBar = new QMenuBar(MainWindow);
        menuBar->setObjectName(QStringLiteral("menuBar"));
        menuBar->setGeometry(QRect(0, 0, 100, 50));
        MainWindow->setMenuBar(menuBar);
        mainToolBar = new QToolBar(MainWindow);
        mainToolBar->setObjectName(QStringLiteral("mainToolBar"));
        MainWindow->addToolBar(Qt::TopToolBarArea, mainToolBar);
        statusBar->setObjectName(QStringLiteral("statusBar"));
        MainWindow->setStatusBar(statusBar);

        retranslateUi(MainWindow);

        QMetaObject::connectSlotsByName(MainWindow);
    } // setupUi

    void retranslateUi(QMainWindow *MainWindow)
    {
        MainWindow->setWindowTitle(QApplication::translate("MainWindow", "MainWindow", 0));
        hostLineEdit->setText(QApplication::translate("MainWindow", "10.10.100.254", 0));
        portLineEdit->setText(QApplication::translate("MainWindow", "8899", 0));
        label->setText(QApplication::translate("MainWindow", "DevIP:", 0));
        label_2->setText(QApplication::translate("MainWindow", "Port:", 0));
        connectButton->setText(QApplication::translate("MainWindow", "Connect", 0));
        stopButton->setText(QApplication::translate("MainWindow", "Disconnect", 0));
        label_3->setText(QApplication::translate("MainWindow", "Smart-clothing platform for elderly care",
0));
        label_4->setText(QApplication::translate("MainWindow", "Temp:", 0));
        clearButton->setText(QApplication::translate("MainWindow", "Clear", 0));
        quitButton->setText(QApplication::translate("MainWindow", "Quit", 0));
        textButton->setText(QApplication::translate("MainWindow", "Record", 0));
        label_5->setText(QApplication::translate("MainWindow", "Pulse:", 0));
        edit_pulse->setText(QApplication::translate("MainWindow", "000 BPM", 0));
        label_6->setText(QApplication::translate("MainWindow", "Posture:", 0));
        edit_att->setText(QApplication::translate("MainWindow", "Standing", 0));
        edit_lat->setText(QApplication::translate("MainWindow", "GPS nosignal", 0));
        label_7->setText(QApplication::translate("MainWindow", "Longitude:", 0));
        edit_lon->setText(QApplication::translate("MainWindow", "GPS nosignal", 0));
        label_8->setText(QApplication::translate("MainWindow", "Latitude:", 0));
        label_x_23->setText(QString());
        label_x_22->setText(QString());
        label_x_16->setText(QString());
```

```
        label_d50_3->setText(QApplication::translate("MainWindow", "80", 0));
        label_x_3->setText(QString());
        label_t300_3->setText(QApplication::translate("MainWindow", "80", 0));
        label_t300_2->setText(QApplication::translate("MainWindow", "40", 0));
        label_x_27->setText(QString());
        label_t300_6->setText(QApplication::translate("MainWindow", "200", 0));
        label_x_2->setText(QString());
        label_x_14->setText(QString());
        label_x_15->setText(QString());
        label_t300_7->setText(QApplication::translate("MainWindow", "240", 0));
        label_x_5->setText(QString());
        label_x_19->setText(QString());
        label_temp_3->setText(QApplication::translate("MainWindow", "Pulse curve", 0));
        label_t300->setText(QApplication::translate("MainWindow", "280", 0));
        label_t300_4->setText(QApplication::translate("MainWindow", "120", 0));
        label_x_4->setText(QString());
        label_x_21->setText(QString());
        label_x_25->setText(QString());
        label_x_17->setText(QString());
        label_d50_5->setText(QApplication::translate("MainWindow", "160", 0));
        label_x_12->setText(QString());
        label_d50_7->setText(QApplication::translate("MainWindow", "Time/s", 0));
        label_x->setText(QString());
        label_d50_6->setText(QApplication::translate("MainWindow", "Pulse/BPM", 0));
        label_x_26->setText(QString());
        label_d50_4->setText(QApplication::translate("MainWindow", "120", 0));
        label_t300_5->setText(QApplication::translate("MainWindow", "160", 0));
        label_d50_2->setText(QApplication::translate("MainWindow", "40", 0));
        label_x_7->setText(QString());
        label_d50->setText(QApplication::translate("MainWindow", "200", 0));
        label_x_11->setText(QString());
        label_0->setText(QApplication::translate("MainWindow", "0", 0));
        label_x_20->setText(QString());
        label_x_8->setText(QString());
        label_x_6->setText(QString());
        label_x_9->setText(QString());
        label_x_10->setText(QString());
    } // retranslateUi

};

namespace Ui {
```

```
    class MainWindow: public Ui_MainWindow {};
} // namespace Ui

QT_END_NAMESPACE

#endif // UI_MAINWINDOW_H
```

Appendix 3. Hardware circuit diagram