Fall Detection System Based On GPS Tracking And Accelerometer Module

Sanja Mandić, Đorđe Novaković

***Abstract*—This paper gives an insight into a fall detection system, whose main purpose is assistance to the elderly. Hardware components used in the design of this system are described, as well as adequate firmware. The first section provides a description of the EasyPIC PRO v7 development board, the GSM/GNSS click module, and the MPU-6050 sensor, whereas the remaining section describes code implementation.**

***Index Terms*—Fall detection; microcontroller; GPS; accelerometer.**

I. Introduction

Falls are frequent occurrences for the elderly and can be caused by many factors such as a loss of balance, fainting, strokes, heart attacks, visual impairment, muscle weakness, dementia, spinal deformations, etc. One of the many consequences of aging is the inability to respond to sudden changes in body position within a timely manner and maintain balance. This drastically increases the probability of falling. The main idea of this paper was to design a system that would be used in assistance to the elderly. Namely, in the case of the fall, a compatible sensor would detect this distinct change in movement, and location data would be automatically sent to a computer of an accountable person. In order to detect the moment in which the fall occurred, the MPU-6050 sensor was used, and the location data acquisition was done with GSM/GNSS module. These two modules were integrated on the EasyPIC PRO v7 development board.

II. EasyPIC PRO v7 development board

The EasyPIC PRO v7 development board, Figure 1, is produced by MikroElektronika. This development board is manufactured in PCB (Printed Circuit Board) technology. The EasyPIC PRO v7 development board supports both 3.3 V and 5 V power supply and contains two power regulators. There are several ways in which the board can be powered: using the USB (Universal Serial Bus) cable, using the adapter connector, or with the use of the screw terminals. This development board supports around a hundred different microcontrollers, among which is the 8-bit microcontroller PIC18F87K22, manufactured by Microchip, serving as a default microcontroller of the EasyPIC PRO v7 development board. In the realization of this project, we used the PIC18F87K22 microcontroller. The microcontroller is integrated with the MCU card (TQFP – Thin Quad Flat Pack card), which is placed in the MCU socket of the development board. Besides microcontroller, the MCU socket includes the 16 MHz crystal oscillator (which determines the clock frequency), the USB communication lines as well as the jumpers for the USB communication and voltage regulator adjustment. This development board features the on-board programmer – mikroProg, which receives instructions from the computer. Additionally, the on-board programmer has an in-circuit debugger, used for real-time debugging on hardware level. The EasyPIC PRO v7 contains connectors compatible with the ICD2 and ICD3 (Microchip) external programmers. There are a several input/output groups (ports: A, B, C, D, E, F, G, H, and J) which include buttons (input), LED (Light Emitting Diodes - output), pins and the three-state switches witch determine if the pull-up resistor (up position), pull-down resistor (down position) or neither one of them (middle position) will be enabled on a certain pin.

The EasyPIC PRO v7 development board features three mikroBUS sockets, allowing for various click boards to be added. Every mikroBUS socket consists of power pins – 3.3 V, 5 V, GND, communication pins – RX (UART – Universal Asynchronous Receiver-Transmitter receive line), TX (UART transmit line), SCL (I^{2}C – Inter-Integrated Circuit clock line), SDA (I^{2}C data line), CS (SPI – Serial Peripheral Interface chip select line), SCK (SPI clock line), MISO (SPI master input slave output line), and MOSI (SPI master output slave input), and also contains single pins – PWM (Pulse Width Modulation output line), INT (hardware interrupt line), RST (reset pin) and AN (analog pin).

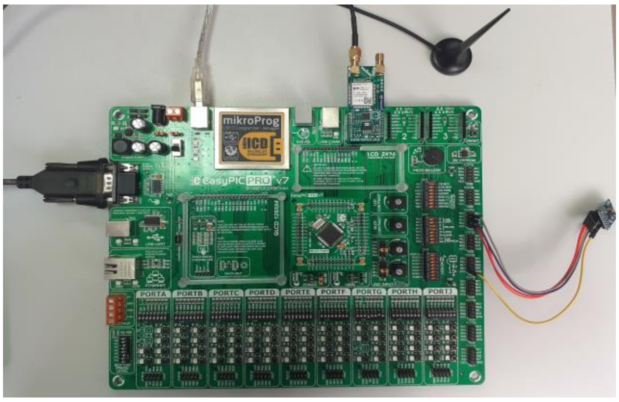


Fig. 1. EasyPIC PRO v7 development board

Sanja Mandić is with the Faculty of Technical Sciences, University of Novi Sad, Trg Dositeja Obradovića 6, 21000 Novi Sad, Serbia (e-mail: sanja.mandic98@gmail.com).

Đorđe Novaković is with the Faculty of Technical Sciences, University of Novi Sad, Trg Dositeja Obradovića 6, 21000 Novi Sad, Serbia (e-mail: djordjenovakovic@uns.ac.rs).

This development board supports the UART communication through the USB connector and through the RS-232 connector. Also, the board contains the USB connector, through witch microcontrollers that support USB communication, can make a connection with other devices.

For the microcontrollers which support the Ethernet communication, there is an RJ-45 connector. The EasyPIC PRO v7 board features an LCD (Liquid Crystal Display) with 2 x 16 character fields, and every field contains a 7 x 5 pixel matrix. The GLCD (Graphical Liquid Crystal Display) with 128 x 64 fields is also included on this board, and touchpanel can be added on the GLCD. Additionally, this development board features a piezo buzzer, an analog temperature sensor, a digital temperature sensor, a reset button, additional GND pins, and a button press level.

A block diagram of the system and connections between the EasyPIC PRO v7 development board and sensors, as well as the PC are presented in Figure 2.

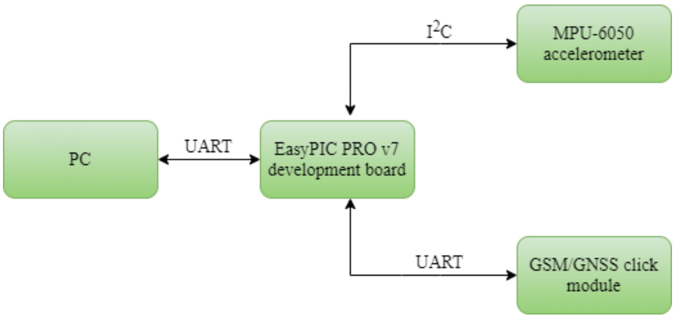


Fig. 2. System block diagram

III. PIC18F87K22 MICROCONTROLLER

The PIC18F87K22 is the default microcontroller on the EasyPIC PRO v7 development board. This microcontroller features nanoWatt manufacturing technology for energy consumption reduction [2, 3]. There are eighty pins; hence, the number of bidirectional ports is nine. The layout of the pins is shown in Figure 3 [2]. The flash program memory has the capacity of 128 KB. The PIC18F87K22 microcontroller includes six 8-bit timers and five 16-bit timers.

The microcontroller features a 12-bit A/D converter, where the number of A/D channels is sixteen. The number of the CCP (Capture-Compare-PWM) modules is ten, three of which are ECCP (Enhanced CCP). CCP modules are peripheral modules for control of various events, and there are three different modes: Capture (a measurement of event duration), Compare (comparison of values of two different registers at the same time), and PWM (pulse width modulation for generating signals) [6].

This microcontroller supports the USART (Universal Synchronous/Asynchronous Receiver-Transmitter), SPI, and I^{2}C communication.

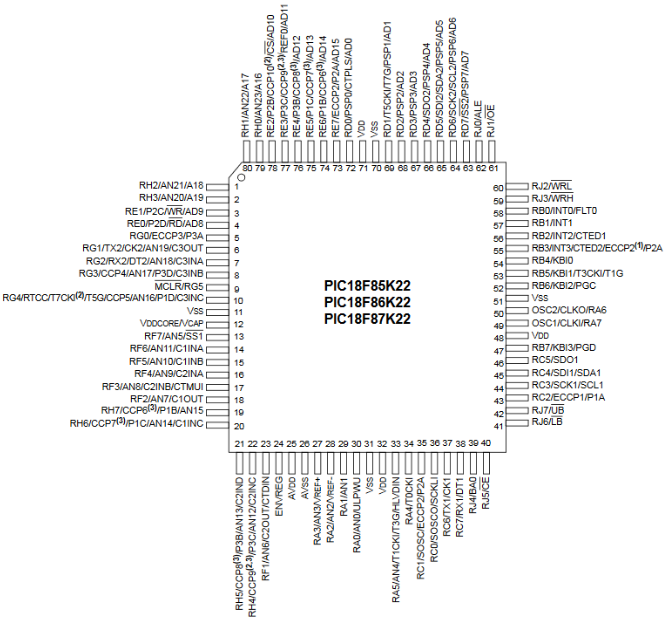


Fig. 3. PIC18F87K22 microcontroller pin layout

IV. GSM/GNSS CLICK MODULE

The GSM/GNSS click module, shown in Figure 4, manufactured by MikroElektronika, is a location tracking module (GNSS), as well as a module for communication with mobile phones (GSM). The location tracking function is achieved based on GPS (Global Positioning System) and GLONASS (Global Navigation Satellite System). The GSM/GNSS click includes the Quetcel MC60 quad-band module, which is the GSM/GPRS module. Communication with a certain microcontroller is realized via the UART protocol. This click board can be powered with 3.3 V and 5 V, and it features Bluetooth antenna, micro SIM and SD card socket, as well as additional pins for connection with the microphone and the speakers.

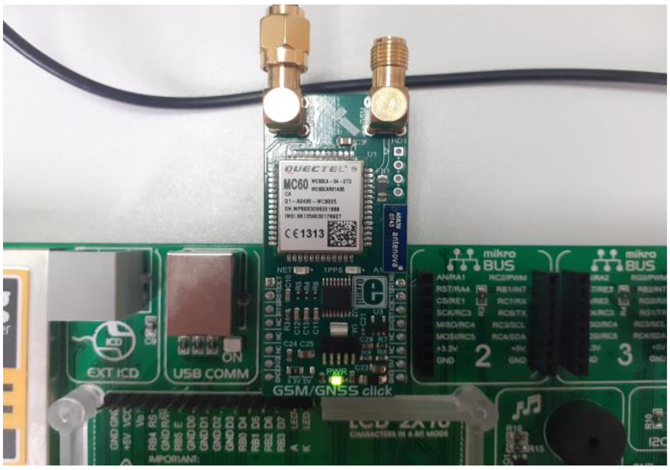


Fig. 4. GSM/GNSS click module

We placed the GSM/GNSS click module in the mikroBUS socket of the EasyPIC PRO v7 development board. Matching pins of this click module and development board are given in Table I.

TABLE I
MATCHING PINS OF THE GSM/GNSS CLICK MODULE AND THE MIKROBUS SOCKET

GSM/GNSS click pin	mikroBUS pin
DET (SD card detection)	AN
RST (reset)	RST
RTS (ready to send)	CS
NC	SCK
NC	MISO
NC	MOSI
3.3V (power supply)	3.3V
GND (ground)	GND
RING (ring indicator)	PWM
CTS (clear to send)	INT
RXD (UART receive)	RX
TXD (UART transmit)	TX
NC	SCL
NC	SDA
5 V (power supply)	5 V

The GSM/GNSS module can perform all GSM (Global System for Mobile Telecommunications) functions, such as making calls and messaging. GNSS is an abbreviation which integrates two satellite systems – GPS (USA) and GLONASS (Russian Federation). This module allows receiving data from satellites, such as coordinates and time. Data is transmitted as encrypted messages, whose format is defined by the National Marine Electronics Association – NMEA. Every line of data that is being transmitted represents a sentence. Every sentence contains specific location data. Some of these sentences are GLL – latitude and longitude, GSA – satellite information, DTM – date, ZDA – date and time, etc. [5]. In order to receive data from satellite, the GSM/GNSS module must be connected to the antenna [4].

V. MPU-6050 SENSOR

The MPU-6050 sensor on the GY-521 board (Fig. 5) integrates the MEMS (microelectromechanical systems) accelerometer with three axes, the MEMS gyroscope with three axes and the DMP (Digital Motion Processor). This sensor uses the I^{2}C communication with a microcontroller (pins: SCL – clock and SDA – data). The MPU-6050 chip has three 16-bit A/D converters for data from each axis. Data from all three axes, x, y, and z, is collected simultaneously. The full-scale range of the gyroscope can be ±250 °/s, ±500 °/s, ±1000 °/s, and ±2000 °/s, while the full-scale range of the accelerometer can take values ±2g, ±4g, ±8g, and ±16g (g = 9.81 m/s^2). The chip contains 1024 byte FIFO (First In Last Out) buffer. Also, the MPU-6050 chip features an oscillator, as well as a temperature sensor.

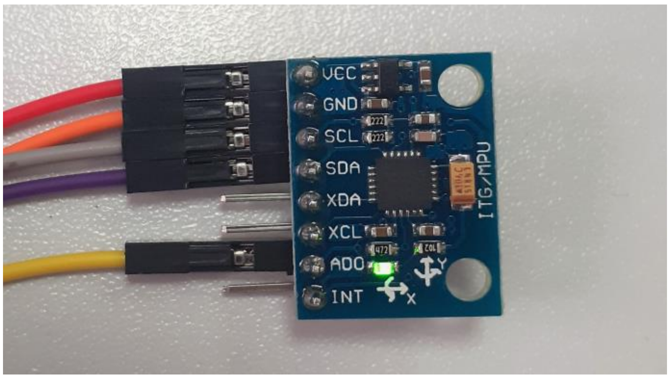


Fig. 5. MPU-6050 sensor (GY-521 module)

TABLE II
MATCHING PINS OF THE GY-521 MODULE AND THE MIKROBUS SOCKET

GY-521 pin	mikroBUS pin
VCC	VCC
GND	GND
SCL	RC3
SDA	RC4
XDA	NC
XCL	NC
AD0	VCC

Figure 6 shows the connection of the GY-521 board and the EasyPIC PRO v7 board. Matching pins of these two components are given in Table II.

AD0 is connected to Vcc to regulate the I^{2}C communication, due to the use of multiple sensors at once.

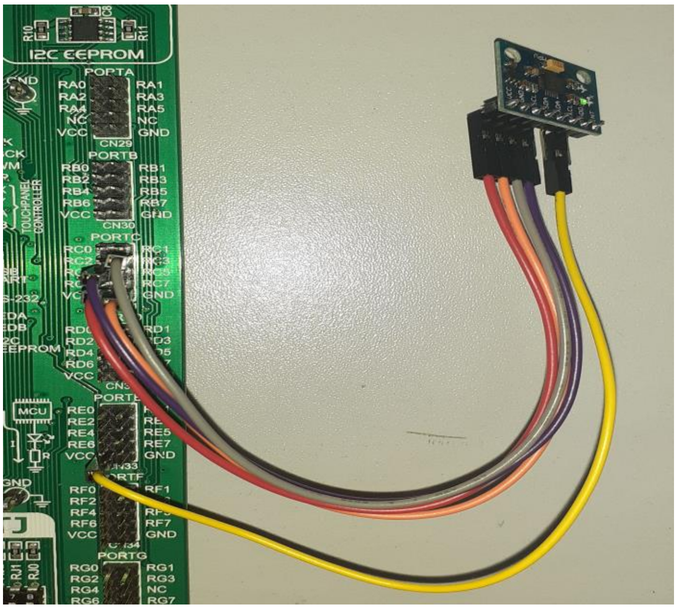
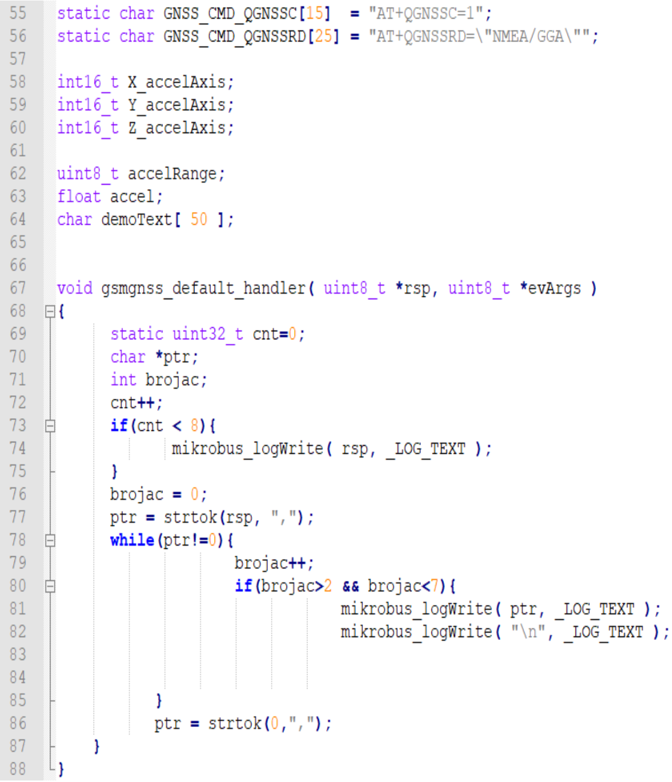


Fig. 6. Connection of the GY-521 and the EasyPIC PRO v7 development board

VI. Firmware

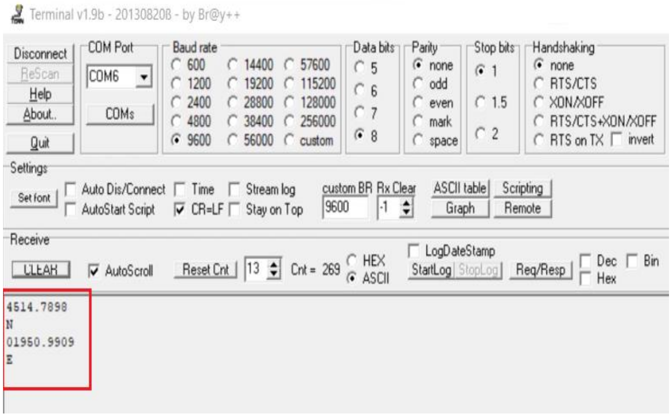
Figure 7 shows the variable initialization. AT commands are used for the MC60 module. For this project, we used *AT+QGNSSC*, power supply control, and *AT+QGNSSRD* for reading location information. *"AT+QGNSSC=1"* is used for turning the module on, whereas *"AT+QGNSSC=0"* means that the module is turned off.

"AT+QGNSSRD=\'NMEA/GGA\'" collects GGA sentences, which contains information about time, latitude, longitude, number of satellites, etc. 16-bit variables *X_accelAxis*, *Y_accelAxis* and *Z_accelAxis* store acceleration values for x, y and z-axis. The variable *accelRange* represents the full-scale range for the accelerometer of the MPU-6050 sensor, while the variable *accel* stores values for the total acceleration. To distinguish latitude and longitude data from the GGA sentence, we parse the GGA sentence. This is done within the function *gsmgnss_default_handler*. The function for parsing the sentence is strtok, which divides given string, in this case, rsp (GGA sentence), into a series of tokens based on the given delimiter, in this case, the comma. Latitude data, information about whether it is north or south, longitude data, and information about whether it is east or west, are contained in places 2-7 in the parsed GGA sentence. Using the *mikrobus_logWrite* function, this data is displayed in the terminal (Fig. 8.)



```
static char GNSS_CMD_QGNSSC[15]  = "AT+QGNSSC=1";
static char GNSS_CMD_QGNSSRD[25] = "AT+QGNSSRD=\"NMEA/GGA\"";

int16_t X_accelAxis;
int16_t Y_accelAxis;
int16_t Z_accelAxis;

uint8_t accelRange;
float accel;
char demoText[ 50 ];


void gsmgnss_default_handler( uint8_t *rsp, uint8_t *evArgs )
{
        static uint32_t cnt=0;
        char *ptr;
        int brojac;
        cnt++;
        if(cnt < 8){
                mikrobus_logWrite( rsp, _LOG_TEXT );
        }
        brojac = 0;
        ptr = strtok(rsp, ",");
        while(ptr!=0){
                        brojac++;
                        if(brojac>2 && brojac<7){
                                mikrobus_logWrite( ptr, _LOG_TEXT );
                                mikrobus_logWrite( "\n", _LOG_TEXT );


                }
                ptr = strtok(0,",");
        }
}
```

Fig. 7. Variable initialization and the GGA sentence parsing



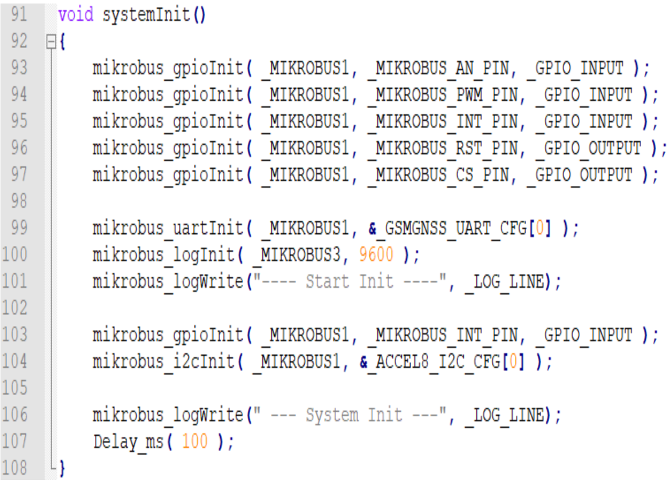
Terminal v1.9b - 20130820ß - by Br@y++

COM Port: COM6
Baud rate: ● 9600
Data bits: ● 8
Parity: ● none
Stop bits: ● 1
Handshaking: ● none
Settings: ☐ Auto Dis/Connect ☐ AutoStart Script ☐ Time ☑ CR=LF ☐ Stream log ☐ Stay on Top
custom BR: 9600 Rx Clear: -1
Receive: ☑ AutoScroll 13 Cnt = 269 ○ HEX ● ASCII ☐ LogDateStamp ☐ Dec ☐ Hex ☐ Bin

```
4514.7898
N
01950.9909
E
```

Fig. 8. Location data

In Figure 9, initialization of the system is shown. *mikrobus_gpioInit* initializes GPIO (General Purpose Input Output) pins as input or output, while *mikrobus_uartInit* initializes the UART communication with the GSM/GNSS module.



```
void systemInit()
{
    mikrobus_gpioInit( _MIKROBUS1, _MIKROBUS_AN_PIN, _GPIO_INPUT );
    mikrobus_gpioInit( _MIKROBUS1, _MIKROBUS_PWM_PIN, _GPIO_INPUT );
    mikrobus_gpioInit( _MIKROBUS1, _MIKROBUS_INT_PIN, _GPIO_INPUT );
    mikrobus_gpioInit( _MIKROBUS1, _MIKROBUS_RST_PIN, _GPIO_OUTPUT );
    mikrobus_gpioInit( _MIKROBUS1, _MIKROBUS_CS_PIN, _GPIO_OUTPUT );

    mikrobus_uartInit( _MIKROBUS1, &_GSMGNSS_UART_CFG[0] );
    mikrobus_logInit( _MIKROBUS3, 9600 );
    mikrobus_logWrite("---- Start Init ----", _LOG_LINE);

    mikrobus_gpioInit( _MIKROBUS1, _MIKROBUS_INT_PIN, _GPIO_INPUT );
    mikrobus_i2cInit( _MIKROBUS1, &_ACCEL8_I2C_CFG[0] );

    mikrobus_logWrite(" --- System Init ---", _LOG_LINE);
    Delay_ms( 100 );
}
```

Fig. 9. System initialization

The *mikrobus_logInit* function defines the UART information logging, where the number of bits that are being transmitted per second is 9600. With this, GSM/GNSS module initialization is achieved. In the *mikrobus_gpioInit* function INT pin of the GY-521 module is initialized as input, and for the I^{2}C communication with the MPU-6050 module initialization, function *mikrobus_i2cInit* is used.

Figure 11 shows the initialization of the application. The initialization of the GSM/GNSS module timer is done in line 116, whereas the initialization of the driver of this module is done in lines 118 and 119. The module is turned on within lines 121 and 122.

The *gsmgnss_cmdSingle* function sends a command to the GSM/GNSS module. Further, in lines 131-138, the driver of the MPU-6050 module is initialized, as well as reset and configure this module. In line 140, the full-scale range of the acceleration is defined, and for this project, we used 8g.

Every 100 ms location data is checked, and in the case of detecting acceleration, which indicates the fall, location data is sent to the user. With the *accel8_getAccelAxis* function values of acceleration for all three axes are collected, and the total acceleration is calculated as given in (1). Projections of the acceleration vector on x, y, and z-axis in the Cartesian coordinate system are illustrated in Fig. 10. This is implemented in line 164 of the code (Fig. 12).

As a limit, we used a value of 30 000. This limit was determined experimentally, by observing acceleration within the normal movement and rapid changes in movement.

$$a = \sqrt{a_x^2 + a_y^2 + a_z^2} \qquad (1)$$

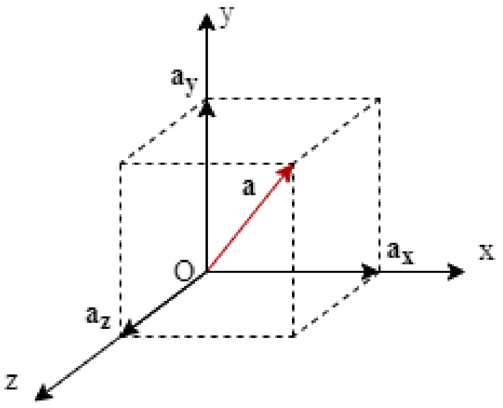
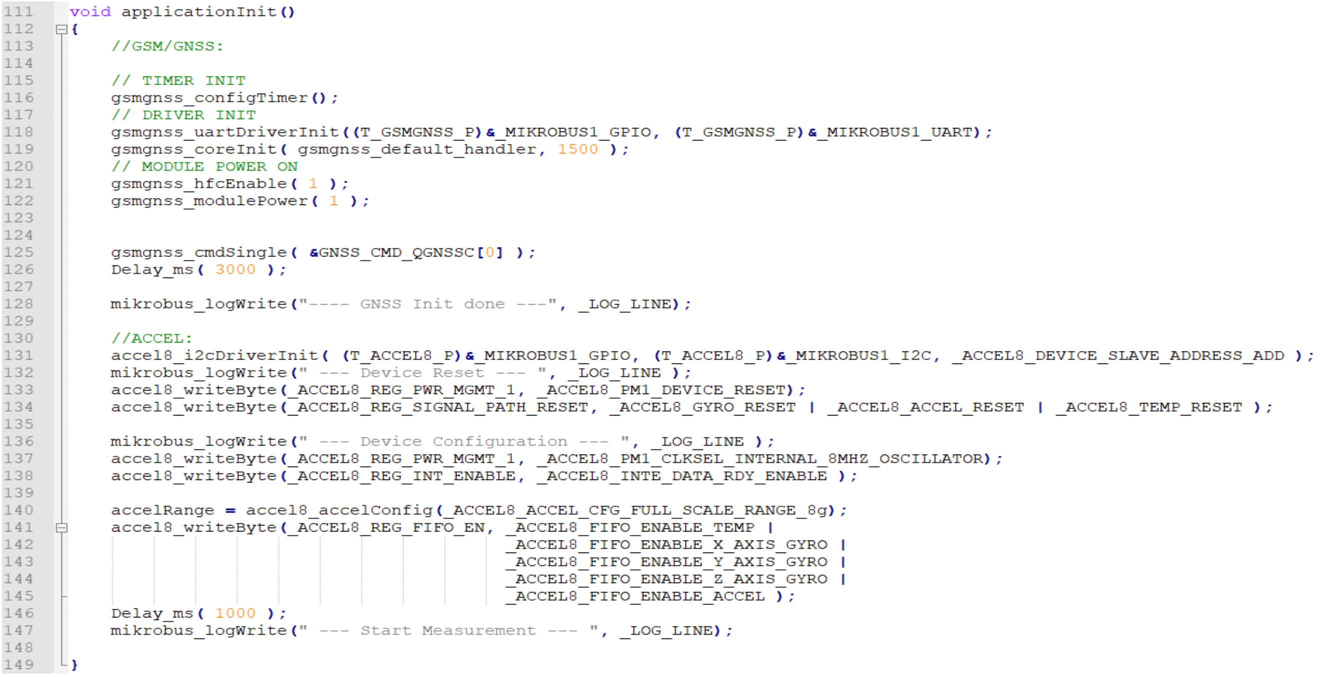
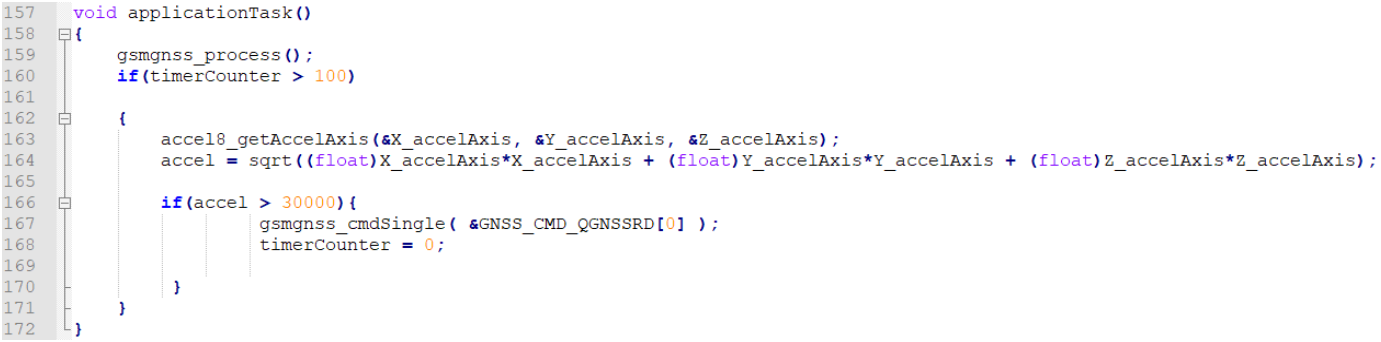


Fig. 10. Projections of the acceleration vector on x, y, and z-axis



```
void applicationInit()
{
    //GSM/GNSS:

    // TIMER INIT
    gsmgnss_configTimer();
    // DRIVER INIT
    gsmgnss_uartDriverInit((T_GSMGNSS_P)&_MIKROBUS1_GPIO, (T_GSMGNSS_P)&_MIKROBUS1_UART);
    gsmgnss_coreInit( gsmgnss_default_handler, 1500 );
    // MODULE POWER ON
    gsmgnss_hfcEnable( 1 );
    gsmgnss_modulePower( 1 );


    gsmgnss_cmdSingle( &GNSS_CMD_QGNSSC[0] );
    Delay_ms( 3000 );

    mikrobus_logWrite("---- GNSS Init done ---", _LOG_LINE);

    //ACCEL:
    accel8_i2cDriverInit( (T_ACCEL8_P)&_MIKROBUS1_GPIO, (T_ACCEL8_P)&_MIKROBUS1_I2C, _ACCEL8_DEVICE_SLAVE_ADDRESS_ADD );
    mikrobus_logWrite(" --- Device Reset --- ", _LOG_LINE );
    accel8_writeByte(_ACCEL8_REG_PWR_MGMT_1, _ACCEL8_PM1_DEVICE_RESET);
    accel8_writeByte(_ACCEL8_REG_SIGNAL_PATH_RESET, _ACCEL8_GYRO_RESET | _ACCEL8_ACCEL_RESET | _ACCEL8_TEMP_RESET );

    mikrobus_logWrite(" --- Device Configuration --- ", _LOG_LINE );
    accel8_writeByte(_ACCEL8_REG_PWR_MGMT_1, _ACCEL8_PM1_CLKSEL_INTERNAL_8MHZ_OSCILLATOR);
    accel8_writeByte(_ACCEL8_REG_INT_ENABLE, _ACCEL8_INTE_DATA_RDY_ENABLE );

    accelRange = accel8_accelConfig(_ACCEL8_ACCEL_CFG_FULL_SCALE_RANGE_8g);
    accel8_writeByte(_ACCEL8_REG_FIFO_EN, _ACCEL8_FIFO_ENABLE_TEMP |
                                          _ACCEL8_FIFO_ENABLE_X_AXIS_GYRO |
                                          _ACCEL8_FIFO_ENABLE_Y_AXIS_GYRO |
                                          _ACCEL8_FIFO_ENABLE_Z_AXIS_GYRO |
                                          _ACCEL8_FIFO_ENABLE_ACCEL );
    Delay_ms( 1000 );
    mikrobus_logWrite(" --- Start Measurement --- ", _LOG_LINE);

}
```

Fig. 11. Application initialization



```
157 void applicationTask()
158 {
159     gsmgnss_process();
160     if(timerCounter > 100)
161
162     {
163         accel8_getAccelAxis(&X_accelAxis, &Y_accelAxis, &Z_accelAxis);
164         accel = sqrt((float)X_accelAxis*X_accelAxis + (float)Y_accelAxis*Y_accelAxis + (float)Z_accelAxis*Z_accelAxis);
165
166         if(accel > 30000){
167             gsmgnss_cmdSingle( &GNSS_CMD_QGNSSRD[0] );
168             timerCounter = 0;
169
170         }
171     }
172 }
```

Fig. 12. Acceleration data acquisition

VII. Conclusion

Due to falling, many injuries can occur in the elderly, such as fractures of arms, wrists, pelvis, hip, head, internal injuries, and loss of mobility. These injuries can significantly reduce the quality of life of the injured person. The fall detection system could contribute to providing timely medical care to the injured, hence, it could contribute to preventing many injuries and medical complications caused by falling.

The aim of this paper was to give a prototype of the fall detection system. To achieve a more reliable system, further work requires a more elaborate experiment and measurements of acceleration.

Further improvements could include the minimization of the components and their integration in one wearable device. This device could be realized as a bracelet, that can be worn on a wrist, ankle, or attached to a belt. An illustration of such a device is presented in Fig. 13.

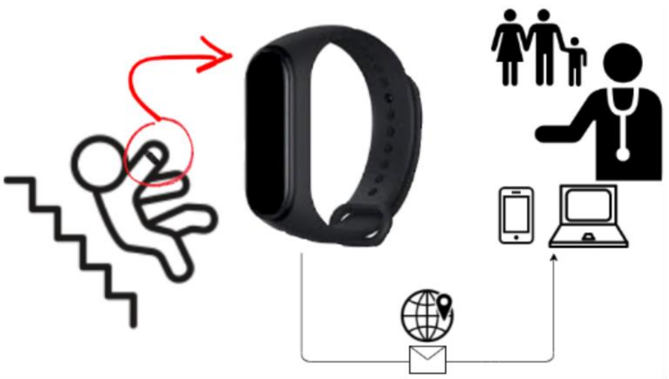


Fig. 13. Fall detection system illustration

References

[1] MikroElektronika. EasyPIC PRO v7 manual. Retrieved February 26, 2020, from https://download.mikroe.com/documents/full-featured-boards/easy/easypic-pro-v7/easypic-pro-v7-manual-v101.pdf

[2] Microchip Technology Inc. PIC18F87K22 Family Data Sheet. Retrieved February 26, 2020, from http://ww1.microchip.com/downloads/en/devicedoc/39960d.pdf

[3] Microchip Technology Inc. nanoWatt Technology. Retrieved February 26, 2020, from http://ww1.microchip.com/downloads/en/Market_Communication/nanowatt1jan03.pdf

[4] MikroElektronika. GSM/GNSS click. Retrieved February 26, 2020, from https://www.mikroe.com/gsm-gnss-click

[5] MikroElektronika (2016). Not all GPS units are created equal. Retrieved February 26, 2020, from https://www.mikroe.com/blog/not-gps-units-created-equal

[6] MikroElektronika. CCP Modules. Retrieved February 26, 2020, from https://www.mikroe.com/ebooks/pic-microcontrollers-programming-in-assembly/ccp-modules

[7] Quectel (2017). MC60 Series GNSS. AT Commands Manual Retrieved February 26, 2020, from https://www.quectel.com/UploadImage/Downlad/Quectel_MC60_Series_GNSS_AT_Commands_Manual_V1.3.pdf

[8] Arduino. MPU-6050 Accelerometer + Gyro. Retrieved February 27, 2020, from https://playground.arduino.cc/Main/MPU-6050/

[9] InvenSens Inc. (2012). MPU-6000 and MPU-6050 Product Specification. Retrieved February 27, 2020, from http://www.haoyuelectronics.com/Attachment/GY-521/mpu6050.pdf

[10] MikroElektronika. Accel 8 click. Retrieved February 27, 2020, from https://www.mikroe.com/accel-8-click