S2AY2526 Assignment

Resources

CG2028_Assignment.zip (https://canvas.nus.edu.sg/courses/85090/files/8487305?wrap=1) ↓ (https://canvas.nus.edu.sg/courses/85090/files/8487305/download?download_frd=1)

CG2028_Assignment_Test.zip (https://canvas.nus.edu.sg/courses/85090/files/8542153?wrap=1) ↓ (https://canvas.nus.edu.sg/courses/85090/files/8542153/download?download_frd=1)

CG2028_Assignment_TestCases.docx (https://canvas.nus.edu.sg/courses/85090/files/8543892?wrap=1) ↓ (https://canvas.nus.edu.sg/courses/85090/files/8543892/download?download_frd=1)

Background

Falls are a major and recurring emergency for older adults. On average, every 32 minutes, an elderly person arrives at a public hospital emergency department due to a fall-related injury [1]. Each month, around 100 seniors are admitted and may remain in the hospital for a week or longer because of these injuries [2]. However, the greatest danger is not always the fall itself; it is the "**Long Lie**", where a person remains on the floor for hours, unable to call for help. This delay greatly worsens outcomes and is associated with a 50% mortality rate within six months.

Your mission is to eliminate the **Long Lie** by designing a device that detects falls, recognises when help is needed, and triggers timely assistance. In other words, a solution that is **smarter than gravity**.

[1] https://www.hdb.gov.sg/about-us/our-role/smart-and-sustainable-living/smart-hdb-town-page/assistive-living-technologies (https://www.hdb.gov.sg/about-us/our-role/smart-and-sustainable-living/smart-hdb-town-page/assistive-living-technologies)

[2] https://ifonlysingaporeans.blogspot.com/2015/07/project-to-reduce-seniors-risk-of.html#:~:text=Year%2Dlong%20scheme%20seeks%20to,an%20injury%20from%20a%20fall (https://ifonlysingaporeans.blogspot.com/2015/07/project-to-reduce-seniors-risk-of.html#:~:text=Year%2Dlong%20scheme%20seeks%20to,an%20injury%20from%20a%20fall)

Overview

Your goal is to use the sensors on the STM32L4S5 Discovery Board to **sense and interpret motion,** so that the device can distinguish a fall from linear movements or bends and respond appropriately. Below is the list of peripherals you should use:

- **Accelerometer**: Measures linear acceleration (G-force). A fall typically shows a brief "free-fall" pattern followed by a sharp **impact spike** at landing.
- **Gyroscope**: Measures angular velocity. It helps detect rapid **orientation changes**, such as a transition from an upright to a lying-down position.
- **LED indicator**: Provides real-time feedback: **fast blinking when a fall is detected**, **slow blinking during normal activity**.
- **(Optional) Barometer**: Measures air pressure to estimate altitude. A rapid pressure change can indicate a **sudden drop in height**, helping separate true falls from actions like sitting down quickly.
- **(Optional) Additional peripherals:** Add extra sensing or response features to improve reliability and usability, such as a **buzzer alarm**, **time-of-flight distance sensing**, **Wi-Fi/Bluetooth communication**, or **data logging** for later analysis.

Detailed Implementation

This project is has two parts:

1. A **pure ARM assembly** moving-average routine in `mov_avg.s`
2. A **C application** in `main.c` that reads sensor data, prints results over UART, and provides a hook for fall-detection + LED behaviour.

Part 1 — `mov_avg.s` (Pure Assembly Moving Average Filter)

`main.c` calls the assembly function with this prototype:
`extern int mov_avg(int N, int* accel_buff);`

So your assembly must follow the ARM calling convention:

- R0: `N` (buffer size)
- R1: `accel_buff` (pointer to an int buffer containing the most recent samples)
- Return (R0): integer average of the `N` samples

Part 2 — `main.c` (Sensor Read → Filter → UART Output → Fall Detection Hook)

At startup, `main.c` initializes the HAL, UART, and sensors:

- `HAL_Init()`
- `UART1_Init()`
- `BSP_LED_Init(LED2)`
- `BSP_ACCELERO_Init()`
- `BSP_GYRO_Init()`

The LED is initially turned off: `BSP_LED_Off(LED2);`

Data buffers and sampling

For accelerometer data, the program keeps three circular buffers (one along each axis), each of length 4.

Each loop iteration:

1. Reads accelerometer raw data into `int16_t accel_data_i16[3]`
2. Writes each axis into its circular buffer using `i % 4`

The program computes filtered acceleration in two ways:

- **Assembly output**
- **Reference C output**

Both results are scaled by `(9.8/1000.0f)` to convert from the sensor's mg-like units into m/s².

UART output

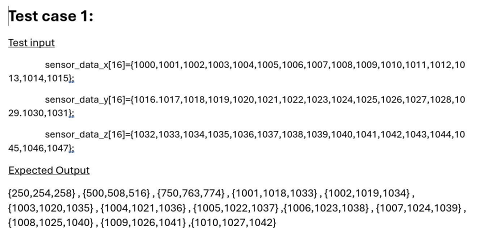
After at least 4 samples have been collected (`if(i >= 3)`), the program prints:

1. Filtered accelerometer results from the **C** function
2. Filtered accelerometer results from the assembly function
3. The gyroscope readings

All printing is done via `sprintf()` into a buffer, then `HAL_UART_Transmit()`.

Procedure

Extract CG2028_Assignment_Test.zip (https://canvas.nus.edu.sg/courses/85090/files/8542153?wrap=1) ↓ (https://canvas.nus.edu.sg/courses/85090/files/8542153/download?download_frd=1) and import it to the IDE. Write your assembly program in the file `mov_avg.s` . Run the project to test your assembly code and compare the output with the reference C function. The test cases along with expected output after each iteration are provided within the {} in CG2028_assignment_TestCases.docx. (https://canvas.nus.edu.sg/courses/85090/files/8543892?wrap=1) ↓ (https://canvas.nus.edu.sg/courses/85090/files/8543892/download?download_frd=1)



Test case 1:

Test input

sensor_data_x[16]={1000,1001,1002,1003,1004,1005,1006,1007,1008,1009,1010,1011,1012,1013,1014,1015};

sensor_data_y[16]={1016.1017,1018,1019,1020,1021,1022,1023,1024,1025,1026,1027,1028,1029.1030,1031};

sensor_data_z[16]={1032,1033,1034,1035,1036,1037,1038,1039,1040,1041,1042,1043,1044,1045,1046,1047};

Expected Output

{250,254,258} , {500,508,516} , {750,763,774} , {1001,1018,1033} , {1002,1019,1034} , {1003,1020,1035} , {1004,1021,1036} , {1005,1022,1037} ,{1006,1023,1038} , {1007,1024,1039} , {1008,1025,1040} , {1009,1026,1041} ,{1010,1027,1042}

For example after the first iteration of calling the moving average filter the 3 outputs will be {250,254,258}. The output after the 2nd iteration shall be {500,508,516} and so on. Once the assembly code is ready copy over the code in `mov_avg.s` from ' CG2028_Assignment_Test ' to `mov_avg.s` in ' CG2028_Assignment ' project.

Now, extract CG2028_Assignment.zip (https://canvas.nus.edu.sg/courses/85090/files/8487305?wrap=1) ↓ (https://canvas.nus.edu.sg/courses/85090/files/8487305/download?download_frd=1) . Once the assembly code is ready copy over the code in `mov_avg.s` from ' CG2028_Assignment_Test (https://canvas.nus.edu.sg/courses/85090/files/8542153?wrap=1) ↓ (https://canvas.nus.edu.sg/courses/85090/files/8542153/download?download_frd=1) ' to `mov_avg.s` in current project(i.e CG2028_Assignment). Edit your code in the main() prgram to meet the assignment requirements. Here, continuously stream accelerometer sensor samples and push each reading into a circular buffer as it arrives. For smoothing, call the mov_avg assembly routine three times per update to filtered readings the X, Y, and Z axes of the acceleration. Next, determine practical thresholds for both accelerometer and gyroscope data to distinguish normal activity from a fall. Use this classification to set distinct LED blink rates, with a higher frequency indicating a detected fall and a lower frequency for normal operation.

Please feel free to integrate other sensors to make the project more interesting.

Test Cases

There will be 3 open test cases and 3 hidden test cases. The open test cases are shown below:

Test case 1:

Test input
sensor_data_x[16]={1000,1001,1002,1003,1004,1005,1006,1007,1008,1009,1010,1011,1012,1013,1014,1015};

Course chat

sensor_data_y[16]={1016.1017,1018,1019,1020,1021,1022,1023,1024,1025,1026,1027,1028,1029.1030,1031};

sensor_data_z[16]={1032,1033,1034,1035,1036,1037,1038,1039,1040,1041,1042,1043,1044,1045,1046,1047};

Expected Output
{250,254,258} , {500,508,516} , {750,763,774} , {1001,1018,1033} , {1002,1019,1034} , {1003,1020,1035} , {1004,1021,1036} , {1005,1022,1037} ,{1006,1023,1038} , {1007,1024,1039} , {1008,1025,1040} , {1009,1026,1041} ,{1010,1027,1042}

Test case 2:

Test input
sensor_data_x[16]={0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};
sensor_data_y[16]={0.0,0,0,0,0,0,0,0,0,0,0,0,0,0.0,0};
sensor_data_z[16]={0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0};
Expected output
{0,0,0} , {0,0,0} , {0,0,0} , {0,0,0} , {0,0,0} , {0,0,0} , {0,0,0} , {0,0,0} , {0,0,0} , {0,0,0} , {0,0,0} , {0,0,0} , {0,0,0}

Test case 3:

Test input
sensor_data_x[16]={6030,6000,4389,4488,6734,5009,7318,6040,5000,5643,3888,3488,3488,3488,4876,5010};
sensor_data_y[16]={9800,9800,6573,6753,5004,5895,4321,6753,9004,3005,6934,4444,7981,5144,5867,4178};
sensor_data_z[16]={5455,6177,8653,7777,6520,4566,7860,5199,6233,5444,9822,3455,6445,7888,4113,4998}
Expected output
{1507,2450,1363} , {3007,4900,2908} , {4104,6543,5071} , {5402,7032,7281} , {5155,6056,6879} , {5887,5493,6680} , {6275,5493,6036} , {5841,6493,5964} , {6000,5770,6184} , {5142,6424,6674}, {4504,5846,6238} , {4126,5591,6291}

General Clarifications and Hints

- For a project of this nature, it is impossible to have the specifications cover all possible scenarios. We leave it to your discretion to decide how the system should respond in scenarios not covered by the above specifications.
- The exact messages/format displayed on the PC is not important. The format and messages are just suggestions. Feel free to make *reasonable* modifications.
- While you are free to set the various thresholds based on experimentation (i.e., empirically), you should be able to explain them in terms of the physical quantities or units involved.

Assignment Submission

The Assignment Canvas submission deadline is as shown below. The assessment schedule will be uploaded to Canvas. **No late submission is considered, try NOT to do last-minute submissions to prevent traffic congestion. You must submit your exported project folder as a .zip file (Archive instructions given below) and submit your slides as a .ppt file.**

Lab Class	Deadline for Canvas submissions
CG2028-B01 (Thursday AM Session)	9:00 AM, 5-MAR
CG2028-B02 (Thursday PM Session)	2:00 PM, 5-MAR

Please follow the rules here to name your archived project and slides (ONE set of submission per group):

Get your group number from Canvas, and name the project file and slides that are to be submitted (DO NOT zip them together) as the following naming templates:

- **For the Project (only .zip file is accepted):**

Assignment1_Group Number_ Matriculation number of all team members separated by an underscore_Code.zip

Example: Assignment_12_A0213331Z_A0010101Z_Code.zip

- **For the Slides (only .ppt file is accepted):**

Assignment1_Group Number_ Matriculation number of all team members separated by an underscore_Slides.ppt

Example: Assignment_12_A0213331Z_A0010101Z_Slides.ppt

The slides have no page limit, and you don't need to attach the entire code as an appendix. Please include the following in your slides and presentation:

- Overview of your system design;
- Detailed implementation of your code, including your overall logic and functions;
- Enhancement/improvement you have done in addition to the compulsory requirements;
- Problems you have encountered during this assignment and how you managed to solve them;
- (Optional) An Appendix for any feedback or suggestions you would like to give to the teaching team,
- and an Appendix that declares every member's joint and specific individual contributions towards this assignment.

Please prepare a 10-minute presentation and demonstration for this assessment.

Contribution Declaration (Week 8)

It is generally expected that every student contributes his/her fair share of the workload and hence everyone in the group gets the same group component of the assignment marks. **If this is not the case** despite your best effort to balance the workload with your partner(s) throughout the entire assignment duration, please complete the online declaration of contribution through **Canvas > CG2028 > Quizzes > Assignment Peer Review** after the assessment. The deadline for online declaration is **12PM (noon) 13-MAR**. The results from the online Contribution Declaration will be cross-checked with inputs from the assessing GA, and the group component for every member will then be adjusted accordingly. Your inputs will be treated with confidentiality and only accessible by the Teaching Team – your partner will not be able to see your comments or how you rated them, so you can and should be honest. The released marks also do not reflect the individual or the group components of the assignment marks.

Last but not least, every team member is reminded to consciously contribute in both joint and individual efforts toward the results of this assignment and declare them clearly in the report's Appendix.

Archive project for submission

1. Go to "File" > "Export".

image.png

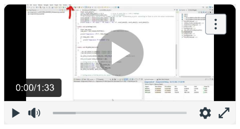
2. Select "Archive File", then "Next >".|

image.png

3. Check the project to be archived. Specify the path and name of the archive file. Save in zip format. Click "Finish".

image.png

Manually Overwrite Sensor Readings For Debug/Demo



Archive Project, Load Archived Project, and Delete Project