November 24, 2016

Thanh Huynh – 2688093

Bilal Butt – 2688700

Michel Clerger – 2694646

Zirui Song - 2652528

GROUP MEMBERS

process report

# General plan and working environment set up

We are planning to do this project basing on the suggestion on LAB Manual. However, at the first week, we experience more with the class diagram and state diagram. Every week, the tasks for individuals will be set up.

|  |  |
| --- | --- |
| week | delivery |
| 1 | (start up)  Test First demonstration  Class diagram (basic)  State diagrams (basic) |
| 2 | Project Initiation Document (including detailed planning) |
| 3 | Specification & implementation Hardware class |
| 4 | Software system design:   * class diagram * state diagrams * sequence diagrams |
| 5 | intermediate demo:   * connection Arduino + Centipede + Laundry Machine demonstrate that the Arduino controls all Laundry Machine features, and that the Arduino monitors all Laundry Machine events * unit tests |
| 6 |  |
| 7 | intermediate demo:   * unit tests |
| 8 |  |
| 9 | final delivery:   * product presentation + demonstration * process description * project documentation |

To sharing and manage code and documents, GIT is used. The repository is set up at: <https://github.com/bilal614/ESP>

# Week 1:

## Group work:

Generally discuss about the group working environment. Investigating the tools for drawing the diagrams such as Microsoft Visio, Astah professional and online tool called https://www.draw.io/. The draw.io is selected since it is an online tools and free, therefore; it is more convenient to use without any installation required

We mainly work on designing the main state diagram and attributes individual tasks.

## Tasks division for implementation

|  |  |  |
| --- | --- | --- |
| Tasks | Responsible members | Status |
| Use draw.io to draw all components of class diagram | Thanh | Done |
| Doing description of the class | Bilal | Done |
| Doing the description for the state diagram and keep track of status document/group discussion. | Michel | Done |
| Redraw the state diagram on draw.io tool | Zirui | Done |

## Assumptions:

We assume that first the user has to select the program that he wants then he can do the payment.

## Questions:

- What should we do for the specification & implementation of Hardware class?

- Should we have a general state diagram or should we also have a very detailed state diagram for each

- Should we have an end state? Or after the main wash it can just go to initial state?

- Should the association between the LaundryMachine class, ProgramExecutor and Program setting are composition associations?

# Week 2&3:

## Group work:

We discuss about some doubts that we had about the interfaces. Then we make sure that everyone have the hardwares that are necessary for our work. We separate the workload between each other.

We mainly work on designing the main state diagram and attributes individual tasks.

## Tasks division for implementation

|  |  |  |
| --- | --- | --- |
| Tasks | Responsible members | Status |
| Implementation and testing of interfaces called “IProgram” and “ICoin” | Thanh | Done |
| Implementation and testing of interfaces called “ILock” and “ISoap” | Bilal | Done |
| Implementation and testing of interfaces called “IWater” and “IMotor” | Michel | Done |
| Implementation and testing of interfaces called “ITemperature” and “IBuzzer” | Zirui | Done |

## Questions:

- Question are indicated on the design document.

# Week 4:

## Group work:

We discuss about what we want to include in the Class Diagram and which additional variable might be necessary for our classes. We have decided to work for now on three classes: ProgramSelect, ProgramSettings and CoinWallet. We start discussing on the ProgramExecutor class and what will be the function of each function that it contains. In the end, we separate the workload between each other.

We mainly work on designing the main state diagram and attributes individual tasks.

## Tasks division for implementation

|  |  |  |
| --- | --- | --- |
| Tasks | Responsible members | Status |
| Implementation of the Final Class Diagram and work on the Class CoinWallet | Thanh | Done |
| Work on the Class ProgramSettings | Bilal | Done for program indicator and cost |
| Modified IWater and IMotor | Michel | Doing |
| Modified Temperature and IBuzzer | Zirui | Done |

## Questions:

- Do we have to display the amount of money left? If so, how can we indicate a remainder of 40, if we have only 3 LED of 10 coin?

- What is expected from the InstallStartHandler() function?

# Week 5:

## Group work:

We discuss about what the intermediary classes. We defined the intermediary classes that we will need, and then we define the different functions that will be needed to make those classes correspond to the requirements of the washing machine. In the end, we separate the workload between each other, everyone should implement the intermediate classes that relates to the interfaces he/she previously created.

## Tasks division for implementation

|  |  |  |
| --- | --- | --- |
| Tasks | Responsible members | Status |
| Implementation the intermediate class related to the interfaces called “IProgram” and “ICoin”, and update the Class Diagram | Thanh | Done with ICoin  Working on IProgram |
| Implementation the intermediate class related to the interfaces called “ILock” and “ISoap” | Bilal | Done |
| Implementation the intermediate class related to the interfaces called “IWater” and “IMotor” | Michel | Doing |
| Implementation the intermediate class related to the interfaces called “ITemperature” and “IBuzzer” | Zirui | Doing or Done? |

## NOTES:

- For the washing machine, should it be able to return change or not?

- Enumeration type for the program? Where should it be? It could be in the ProgramSettting?

- Is it possible to declare an object of interface class in C++? ILock lock or ILock \*lock? We are planning to do it in the pointer way.

- What does the ‘add soap 1’ mean? Is it different from the ‘closed soap compartment’?

- Why in the ProgrameSelect Constructor we need to be provided a pointer to IProgram in its argument

-Why is there an output pin for lock when there is no corresponding LED or other hardware component for it?

# Week 6:

## Group work:

* Compiled CoinWallet, ProgramSelect and Soap and Door via the Program executors and debugging the problems
* Discuss about InstallStartHandler to get the working concepts of it.

## Tasks division for implementation

|  |  |  |
| --- | --- | --- |
| Tasks | Responsible members | Status |
| Implementation the IProgame class. Investigate and implement the StubHardwareControl.cpp functions used for testing the CoinWallet Class  Sequence diagram for IsReady() function in ProgramExecutors  Stated diagram for CoinWallet class | Thanh | Having problems with setting environments for running the test.  Done with the rest of tasks |
| Work on the ProgramExecutor class  Making descriptions of Soap and Lock classes | Bilal | Doing ProgramExecutor  Done with description |
| Continuing on the immediate classes Motor and Water. Introducing state diagram in case of it is necessary.  Making descriptions of Motor and Water class | Michel | Doing |
| Continuing implement the ProgramSetting classes.  Making descriptions of Temperature and Buzzer classes. | Zirui | Done |

## NOTES:

- Try to install Codeblocks to run the unit test on Window environment. However there was a compile a error when doing the compilation is done. This problem is solved by install mingw.org <http://www.mingw.org/category/wiki/download>

- Then it can be compiled, but some data types like char or byte are not recognized by Codeblocks even the compiler is selected is AVR GCC

- When we try to run functions of CoinWallet class, the soap and lock together in the program executor via the main loop(). It turns out some unknown behaviors.

# Week 7:

## Group work:

* Compiled CoinWallet, ProgramSelect and Soap and Door via the Program executors and debugging the problems
* Discuss about InstallStartHandler to get the working concepts of it.

## Tasks division for implementation

|  |  |  |
| --- | --- | --- |
| Tasks | Responsible members | Status |
| Implementation the IProgame class. Investigate and implement the StubHardwareControl.cpp functions used for testing the CoinWallet Class  Sequence diagram for IsReady() function in ProgramExecutors  Stated diagram for CoinWallet class | Thanh | Having problems with setting environments for running the test.  Done with the rest of tasks |
| Work on the ProgramExecutor class  Making descriptions of Soap and Lock classes | Bilal | Doing ProgramExecutor  Done with description |
| Continuing on the immediate classes Motor and Water. Introducing state diagram in case of it is necessary.  Making descriptions of Motor and Water class | Michel | Doing |
| Continuing implement the ProgramSetting classes.  Making descriptions of Temperature and Buzzer classes. | Zirui | Done |

## NOTES:

- Introduce the static variable in the hardware class to keep track the sharing data between the controlling of LEDs for coin 200 and the soap 2