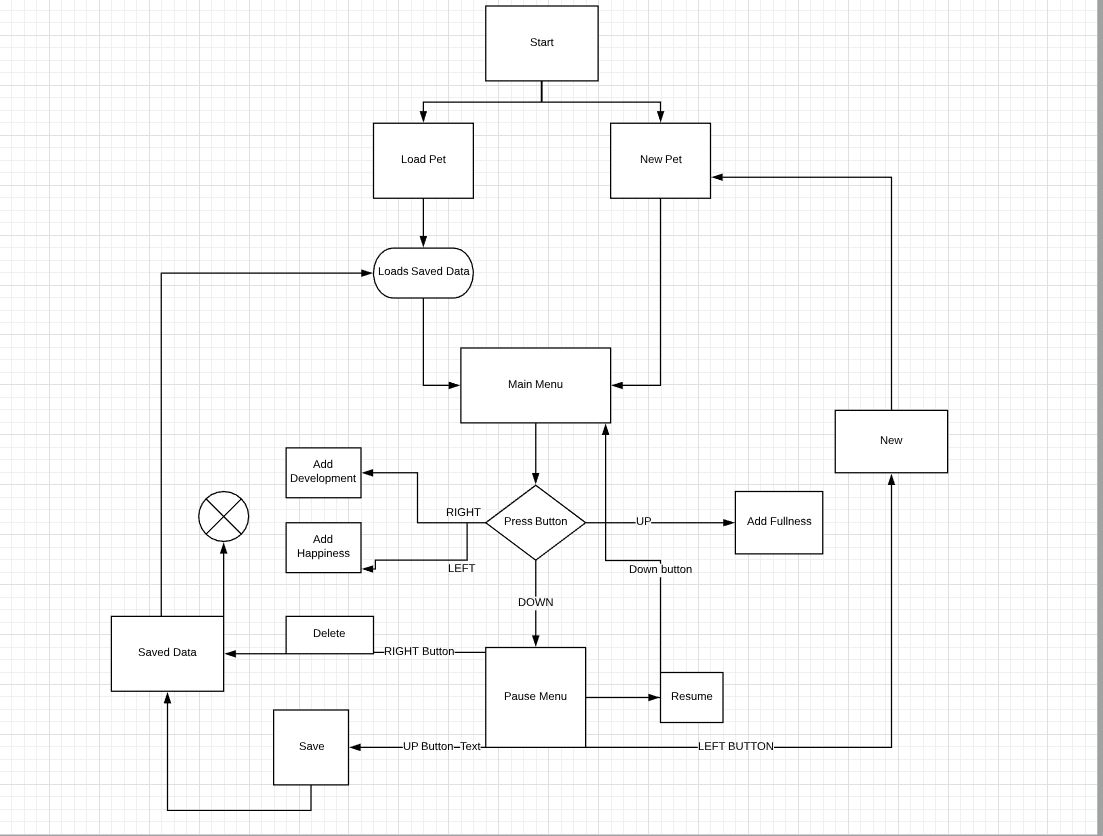
Embedded Systems Coursework Report

**Requirements**

1. Develop a software that simulates a virtual pet.
2. Check whether a saved virtual pet from a previous run is present.
3. Display statistics of virtual pet.
4. Apply certain conditions and rules on pet depending on values of statistics.
5. Include feature of pausing the simulation.
6. Include feature that saves, deletes and creates a virtual pet.

**Design and User Guide**



The first menu the user is shown is the startup menu. This gives a choice for the user to either load a virtual pet that has been saved before or create a new pet. Both result in the menu switching to the main menu. In the case of the new pet made the app shows the starting statistics of the pet which include (fullness, happiness, age and development). In this menu the user can interact with the virtual pet. By meeting certain conditions the user can increase the development by pressing right button, adding happiness by pressing left button, adding fullness with down button and last but not least pause the simulation but adding the down button. In the pause menu the user is given options of saving the virtual pet by saving its statistics so next time its loaded the statistics are as one saved. The delete button allows to delete any previously saved pet. The new button will essentially restart the app by making a new virtual pet ( this still allows for a saved pet to be loaded). Finally the resume button brings back the user to the main menu and allows unpauses the simulation.

**Implementation**

First step is to include all necessary libraries that will be used later on in the simulation. These include time libraries and eeprom libraries for the save functionality. A structure is then created for the pet that contains all of its statistics (development, happiness, fullness, age). In the setup stage I use EEPROM.get to retrieve the statistics of the pet. Then I implemented a while loop for the startup menu that prints out necessary prompts to the user and gives the option of pressing the up and down buttons to start up the main simulation. An important detail in these is a variable that allows age to keep working even when a pet is loaded by introducing a variable that add a sort of counter to the age. This keeps the age in check whether a new pet is created or an old is loaded. If a new pet is loaded the starting statistics are implemented and user is moved to a new main menu. By setting the cursor to different areas of the screen, the statistics of the virtual pet are clearly shown. In the loop section, each of the button are given select jobs using if statements. After meeting the necessary conditions such as every 7 seconds happiness is reduced by 1, the virtual pet has its statistics change over time. The timing of these changes is controlled via millis function. By converting to seconds and using the previously mentioned controlling variables the conditions can be met by using millis on the age of the pet and then using the age to decide whether to reduce or increase the rest of the statistics. Delays are added to the button in main menu to refrain the simulation from crashing and by stopping the user from spamming the buttons. This delay isn’t big enough though to make the simulation feel tedious. Also for every button that requires user to switch screens a clear screen function is used so that new information can be displayed on the LCD. After pressing the pause menu the simulation is paused by having the timer stop which as discussed before was in control of the rest of the statistics. After reaching new menu more complicated functions are available. The simplest of those is the resume function/button which when pressed resumes the timer and returns user to main menu screen keeping the simulation running as before. The new function simply clears the screen and after a delay sets the initial starting statistics to the virtual pet and the timer variables to refrain to keep the age in control of the other statistics. The save button “saves” as the name suggests the virtual pet to EEPROM, where the pets statistics are recorded and a screen is shown giving user a message that pet is saved after which user is returned back to pause screen. Finally the delete button removes all data of a virtual pet that has been previously saved to EEPROM and also giving user a warning message that the pet has been deleted. If a pet is saved that will allow the user during startup menu to load that pet with all its saved statistics and continue from that point on.

**Testing**

Test 1: On starting check whether the software checks if a saved virtual pet is present in EEPROM and allowing user to load a pet or create a new one.

Result: 

Test 2: Display all the statistics of virtual pet on LCD.

Result: 

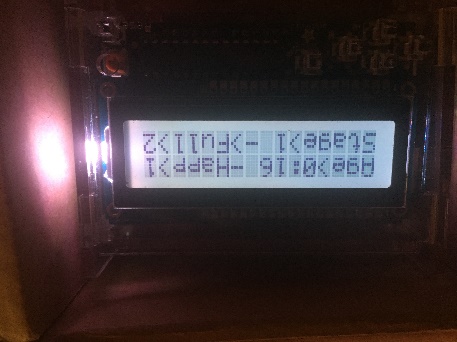
Test 3: When age reaches 5 seconds, development stage is set to 1.

Result: 

Test 4: Every 7 seconds, happiness is reduced by 1.

Result: 

Test 5: Every 11 seconds, fullness is reduced by 1.

Result: 

Test 6: Save the pet.

Result: 

Test 7: Delete a saved pet.

Result: 