CS/EE 120B Custom Laboratory Project Report

Tamagotchi !!

Anthony Padilla

06/13/2024

https://youtu.be/BvmTBJO7Yw0

Introduction

Inspired by the classic tamagotchi that took the world by storm, this rendition will attempt to recreate classic functionality of a virtual pet, such as feeding the pet to keep it full, and playing with the pet to keep it happy. There are multiple games that can be played with the pet, and different food options to give it.

The pet hatches after a certain amount of time, depending on certain factors, specifically games played. Then, a new pet(sprite) will appear to be taken care of ! If left unattended, the pet will in fact die :(.

All aspects of the project were implemented successfully. The pet hatches, can be played with, properly displays status, enjoys food, and all games work properly. However, due to how complex this project is (essentially being multiple games and a large system), it got out of hand quickly. Because of this, some fixes were like band-aids to bug fix on a time crunch. Usability is not limited, but I believe in a second version, with reduced SM design and now knowing what I learned on this project, I could make it essentially not only bug free, but very clean and smooth. Many small bugs are mainly out of sync music playing/stalled music, and conflicts with the buzzer.

Build-upons

List the three (or more) build-upons that you have implemented. Clearly distinguish between the build-upons that were implemented successfully, and those that were not fully implemented, or were not implemented at all.

The buildupons I implemented were the use of an **LCD screen, Passive Buzzer, and having 2 pets.** I was unable to implement clicker gameplay as that would require a helper function to be able to display digits properly on the screen, which was a lot considering I already have more than 10 sprites, 3 being detailed with multiple stages.

While I was able to implement EEPROM.h functionality, it is an arduino library, so instead it was an extra thing for personal use. I ran out of time to implement using the avr/eeprom.h

For the LCD Screen, I was able to fully use all color-abilities, clear and update the screen consistently, based on what was happening in game logic.

For the Passive Buzzer, I had a music option that plays during the Ryuk_Bird Minigame, as well as a music button in the Pet Idle state to turn the music off and on. The music playing is the first chorus of the song Solitude by Sakamato ,
https://youtu.be/HBm2YKibeLA?si=j496XN3LhSLgU2zV

And for 2 Pets, the user is now able to manage two pets at a time, toggling the pet with B2.

User Guide

Inputs: B1-B5

Pets: Misa (Deathnote), Kiara (JJK), Alice (Madness Returns)

PetIdle:

The pet will always be in the PetIdle state. Every hour there will be a check which will lower happiness and raise hunger. The user can always check pet status by pressing B3 in pet Idle, which will display the desired food and current status (angry face for hungry, sad face for low happiness, green if all good).

In PetIdle, the user can toggle the music by pressing B4, or reset their pet using B5.

In PetIdle, the user can press B1 to enter the select screen for Pet activity options.

The user can toggle what pet they will interact with by pressing B2. The user can tell what pet is selected by the number in the lower left corner. 1 for pet 1, 2 for pet 2.

Initially, the pet is an egg. The egg will hatch after 24 hours for BOTH pets, with a sprite based on certain conditions. If the user were to somehow read a secret guide or manual, they can confirm what pet to get.

If the pet was checked on and it was the 3rd time the pet was not being taken care of, the pet will die and become and egg again.

PetSelect:

In pet select, the user can toggle their selection using B1, and confirm their selection with B2. If they select the food option, they enter the feeding menu. If they select the game option, they will be taken to the game select option.

The user can press B3 to go back to pet idle

Pet Feeding Menu:

In this menu, the user can toggle their choice with B1 and confirm with B2. Animations will play, confirming their choice (correct or incorrect) and going back to the Idle state.

The user can press B3 to go back to pet idle

Pet Game Menu:

In the game menu, the user can toggle their game choice using B1, and confirm with B3. The basket sprite selects the basket game, the letter selects simon says, and the ryuk head starts the ryuk game.

The user can press B3 to go back to pet idle

Pet Basket Game:

In the basket game, the user must catch 6 apples with the basket. The user can move left with B1 or right with B2. If the user misses more than 2 apples, the game loses. Winning the game increases pet happiness.

Pet Simon Says Game:

In the simon says game, the player is shown a sequence of 2 characters, with A corresponding with B1, and B corresponding with B2. If the player gets 3 sequences correct, they win the game. If they miss one, they instantly lose. Colors are shown to the player when they are correct.

Pet Ryuk Game:

In this game, the player uses B1 to continuously jump over moving ryuk heads. If they survive for a certain amount of time, they win they game. If they are hit by a head, they lose.

*In every game state, corresponding happy and sad sprites will be shown based on winning/losing. Also good/bad noises will play with the buzzer.

Hardware Components Used

- **5 buttons**
- **1 HiLetgo LCD 128*128 screen**
- **1 Passive Buzzer**
- **6 1kOhm resistors**

Software Libraries Used

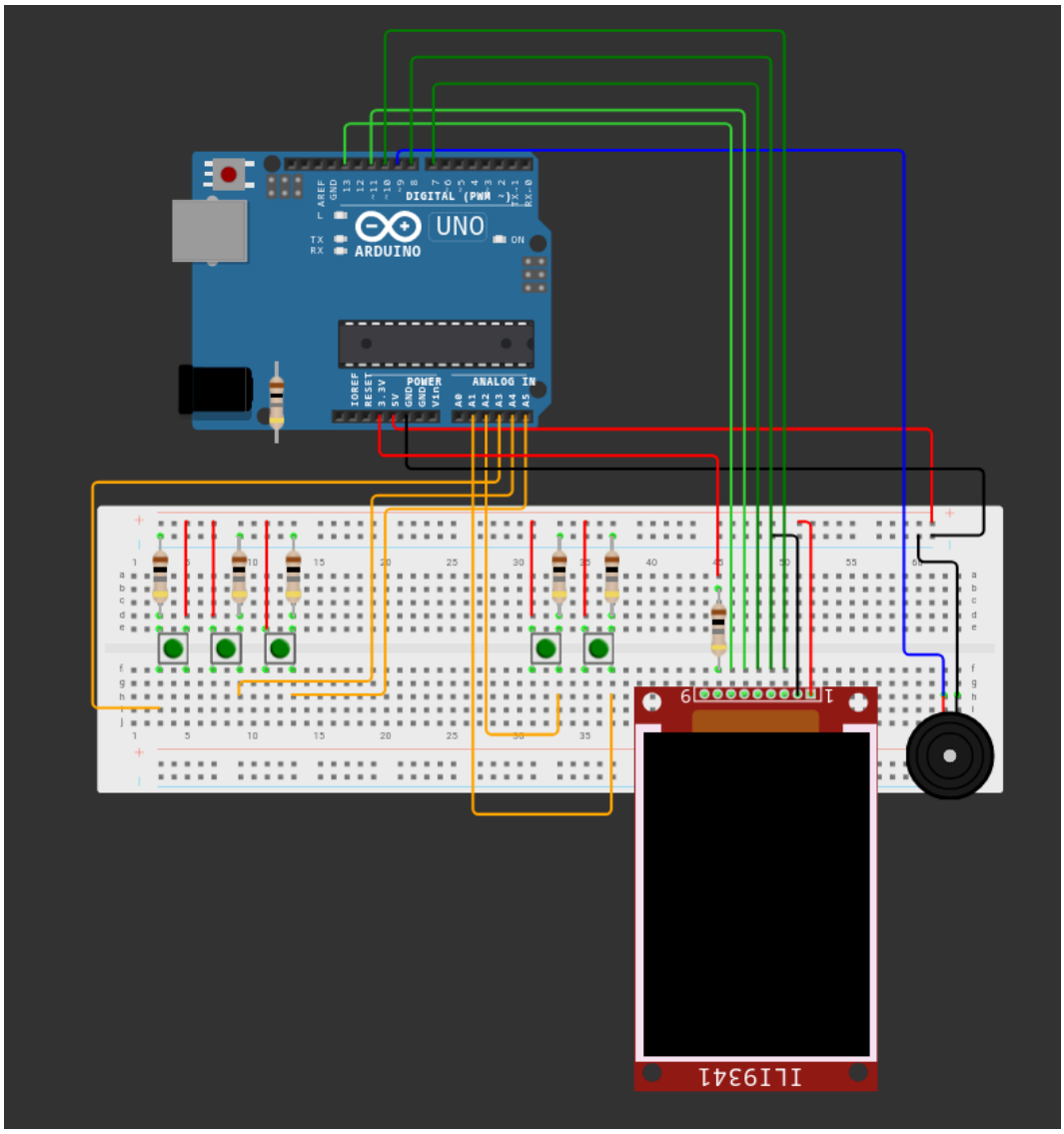
I did not use any software library/header files other than those provided by the TA's or header files of my own.

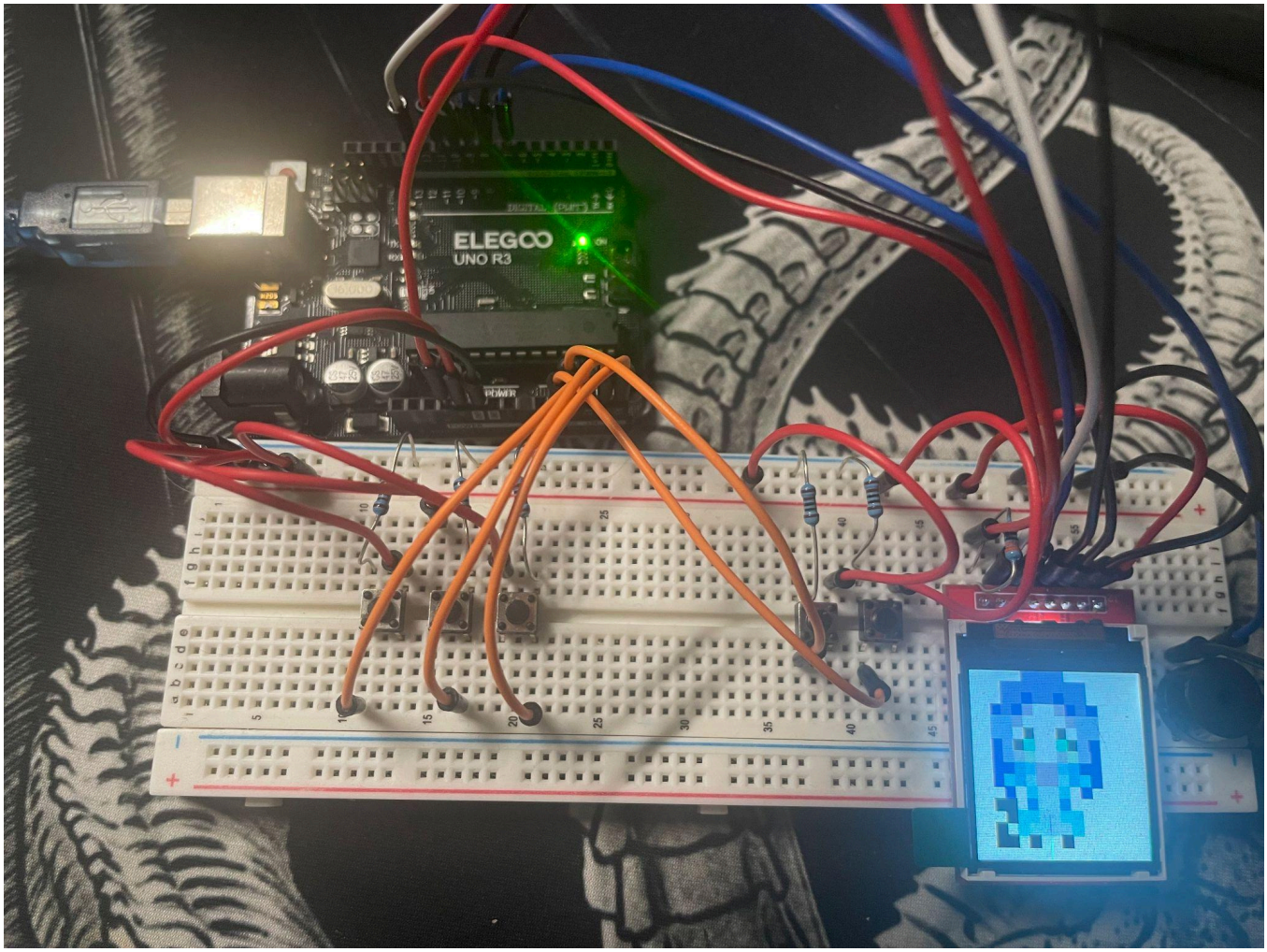
Of my own header files, I have one for each game, 1 for the pet struct, and 1 that controls both the LCD screen initialization and the print function of every sprite, INCLUDING clearing the sprites or altering them.

Wiring Diagram

Draw a wiring diagram to show the physical setup of the system.

(This LCD has 9 pins while the one I purchased has 8, the unused pin is MISO)





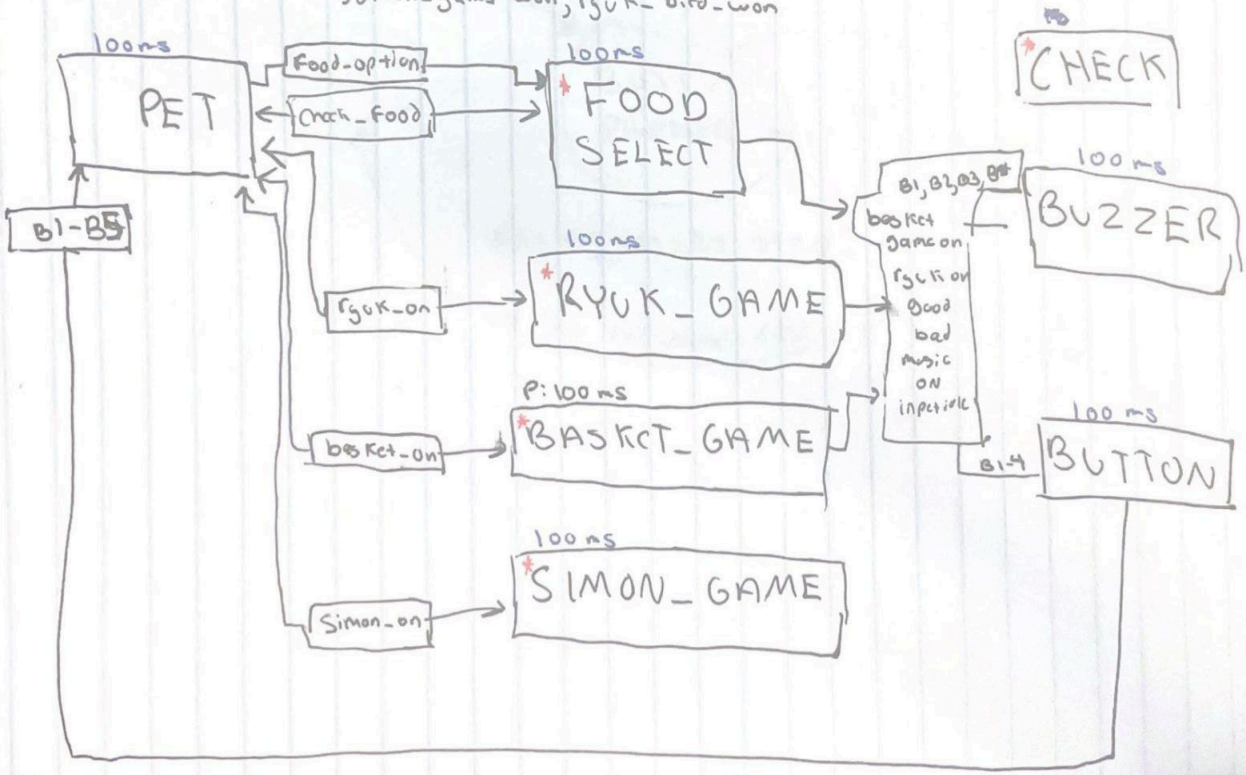
Task Diagram

Global variables:

* Pet Pet, Pet_2

bool displaying_food_switch, displaying wanted, Pet_1_hatched, Pet_2_hatched, in_pet, idle, Pet_selected, music_on, caught, pause, check_food, basket_game_on, ryuk_on, simon_on, good, bad, buzzing, B1-B5 ~~B1, B2, B3, B4, B5~~ (check_food)

int basket_game_won, simon_game won, ryuk_bird_won



* = writes to pet

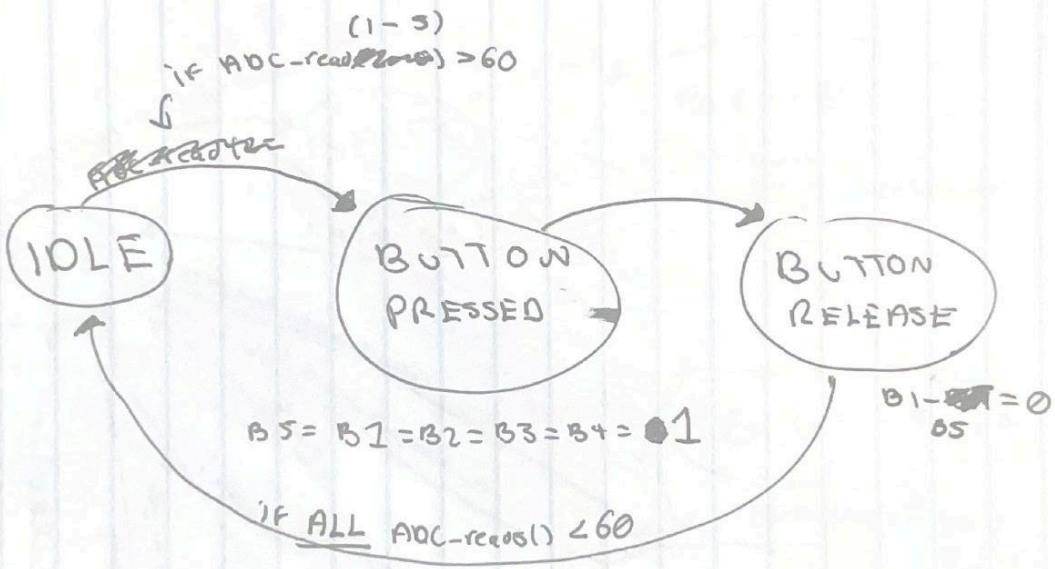
SynchSM Diagram

Sorry its alot

BUTTON SM

bool B1 - B4

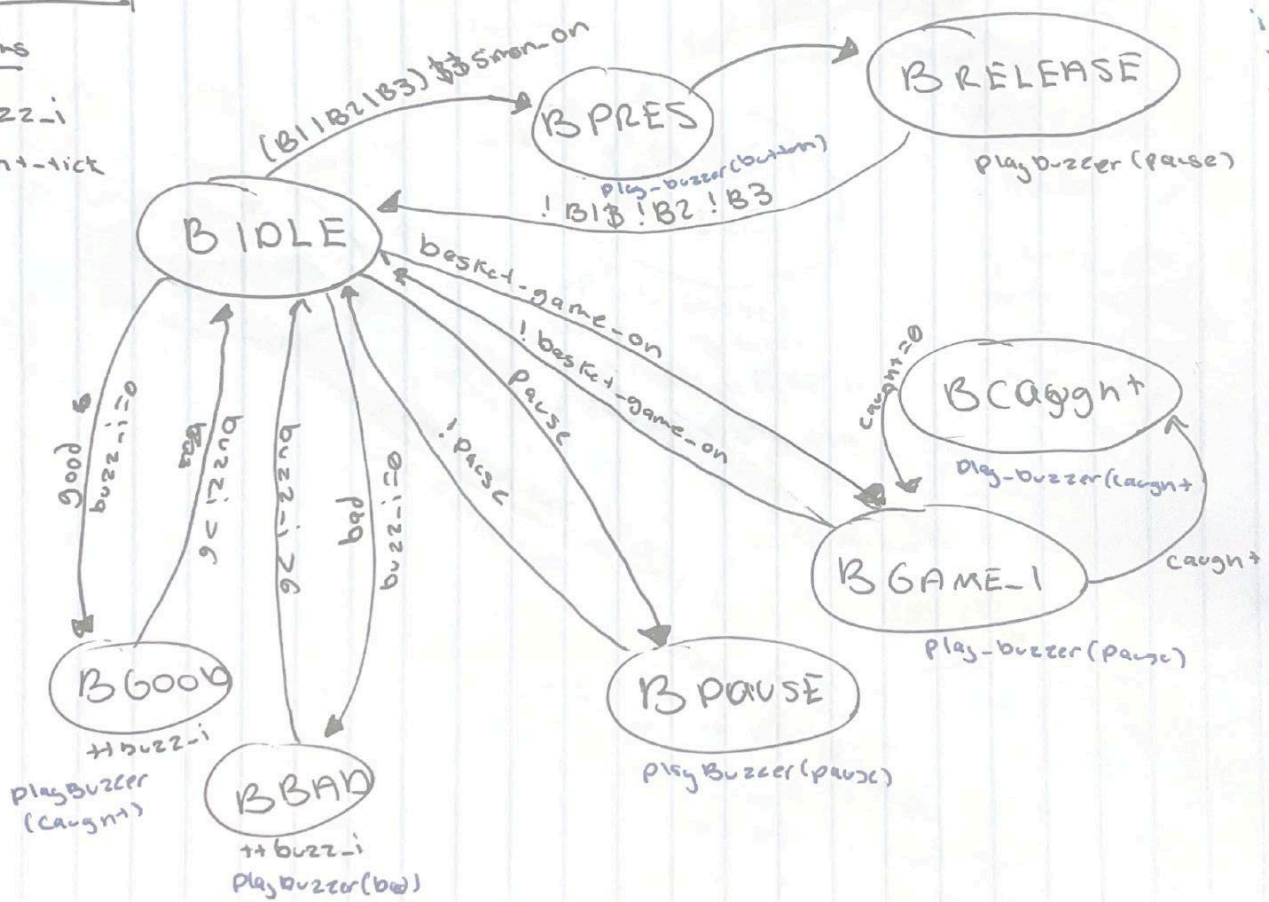
P: 100ms



BUZZER

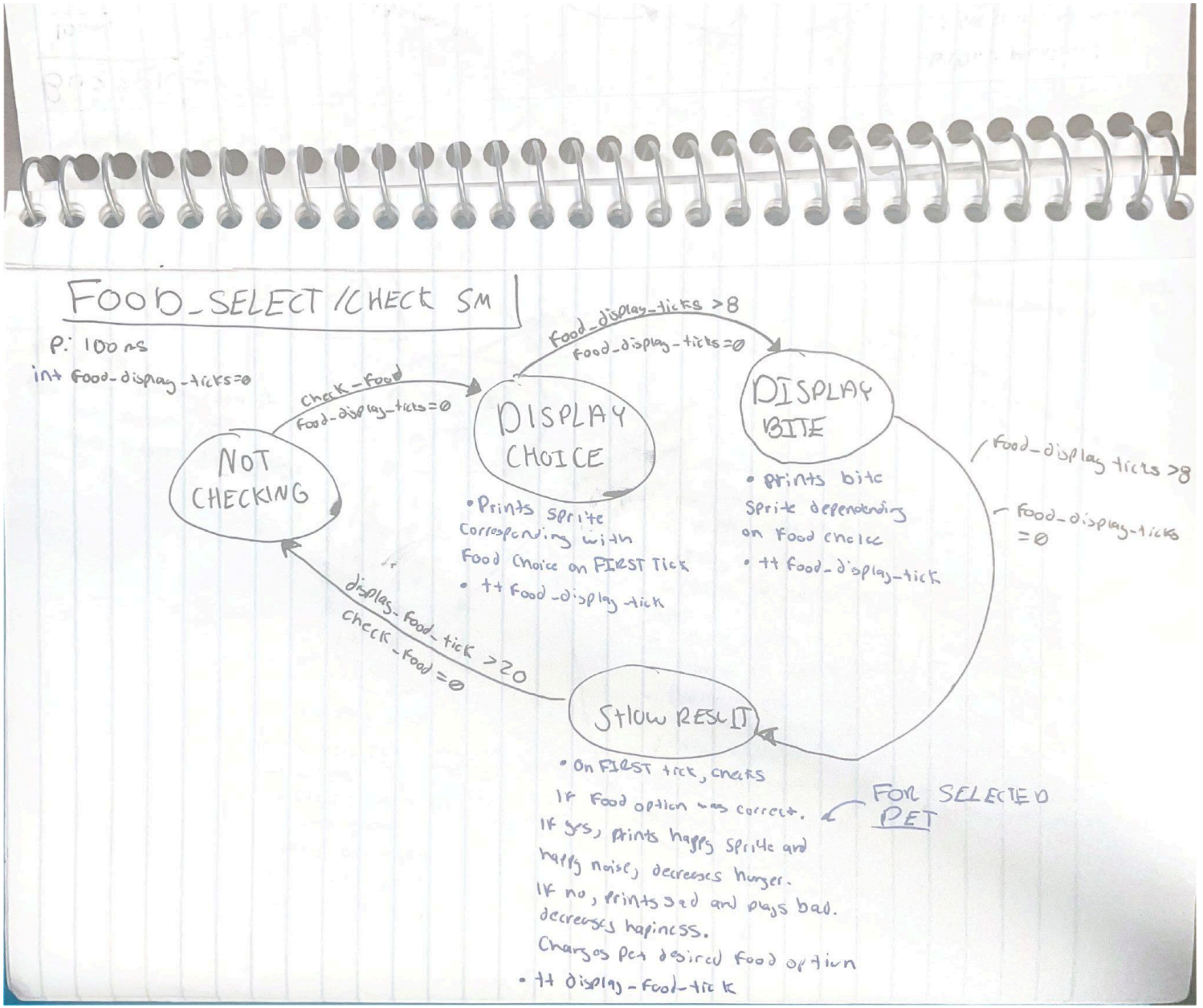
100ms

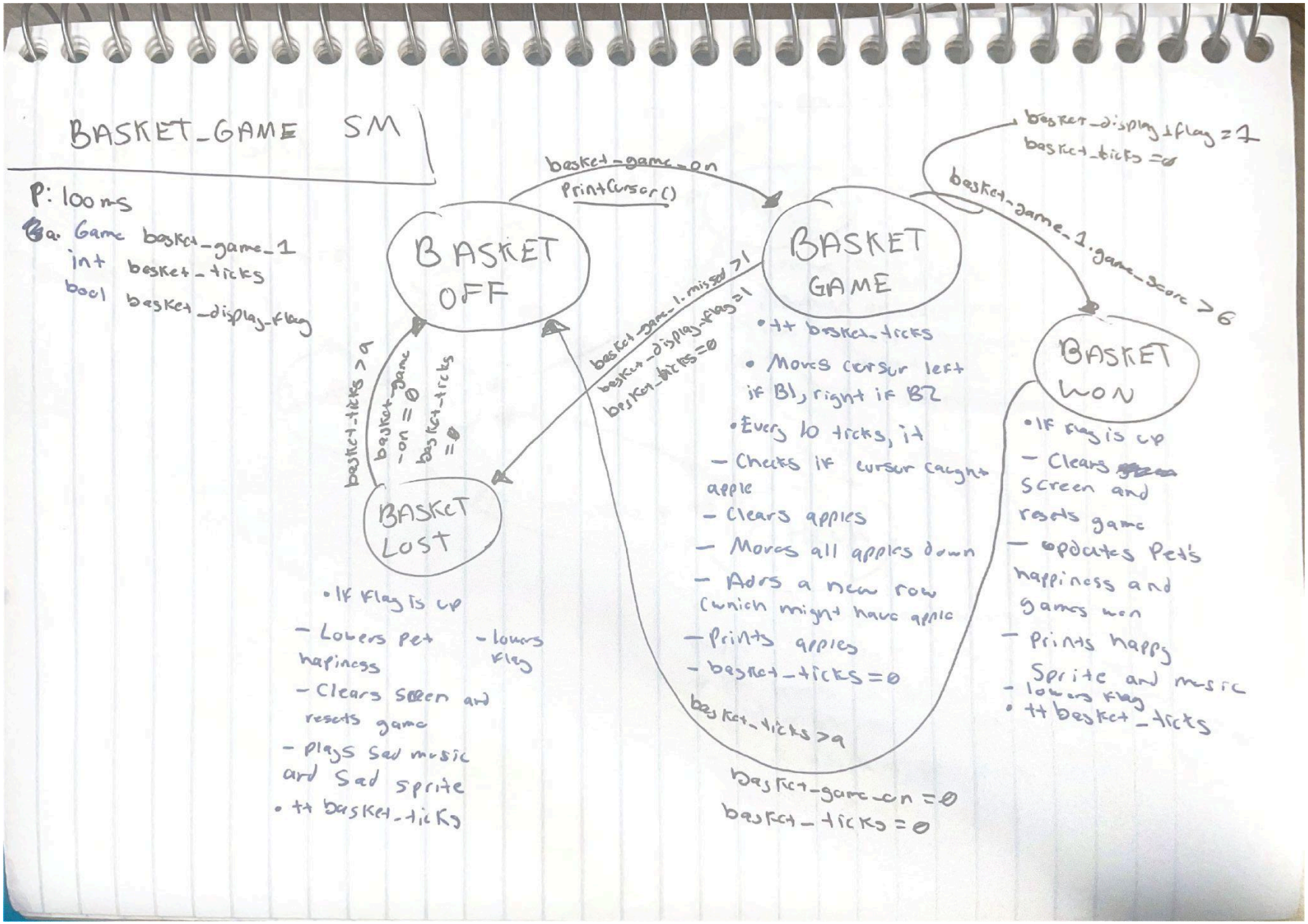
int buzz_i
int caught_tick



BIDLE Action:

if in idle and music is on, play music tone every 5 ticks, or if simon_game is on, pause buzzer

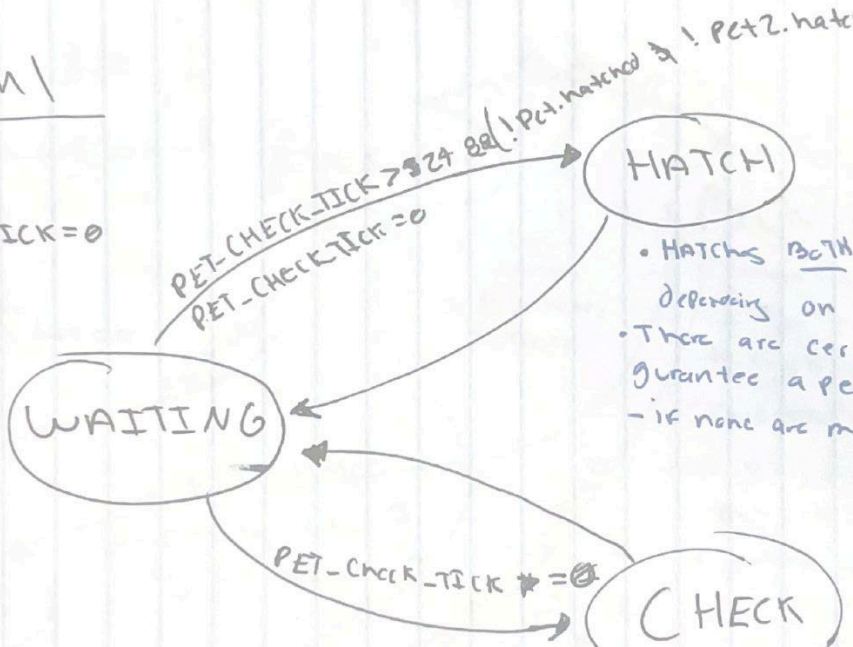

FOOD_SELECT/CHECK SM
P: 100 ms
int food_display_ticks=0
NOT CHECKING
DISPLAY CHOICE
DISPLAY BITE
SHOW RESULT
check_food
food_display_ticks=0
food_display_ticks >8
food_display_ticks=0
food_display_ticks >8
food_display_ticks =0
display_food_tick >20
check_food=0
• Prints sprite corresponding with Food Choice on FIRST Tick
• tt food_display_tick
• Prints bite sprite dependending on food choice
• tt food_display_tick
• On FIRST tick, checks if Food option was correct.
If yes, prints happy sprite and happy noise, decreases hunger.
If no, prints sad and plays bad. decreases happiness.
Changes pet desired food option
• tt display_food_tick
FOR SELECTED PET


BASKET_GAME SM
P: 100 ms
a. Game basket_game_1
int basket_ticks
bool basket_display_flag
basket_game_on
PrintCursor()
BASKET OFF
BASKET GAME
BASKET LOST
BASKET WON
basket_display_flag = 1
basket_ticks = 0
basket_game_1.game_score > 6
basket_game_1.missed > 1
basket_display_flag = 1
basket_ticks = 0
basket_ticks > 9
basket_game_on = 0
basket_ticks = 0
• ++ basket_ticks
• Moves cursor left if B1, right if B2
• Every 10 ticks, it
– Checks if cursor caught apple
– Clears apples
– Moves all apples down
– Adds a new row (which might have apple
– Prints apples
– basket_ticks = 0
• If flag is up
– Clears screen and resets game
– Updates Pet's happiness and games won
– Prints happy Sprite and music
– lowers flag
• ++ basket_ticks
• If flag is up
– Lowers pet happiness
– Clears screen and resets game
– plays sad music and Sad sprite
• ++ basket_ticks
– lowers flag

CHECK SM

P:

int PET_CHECK_TICK = 0



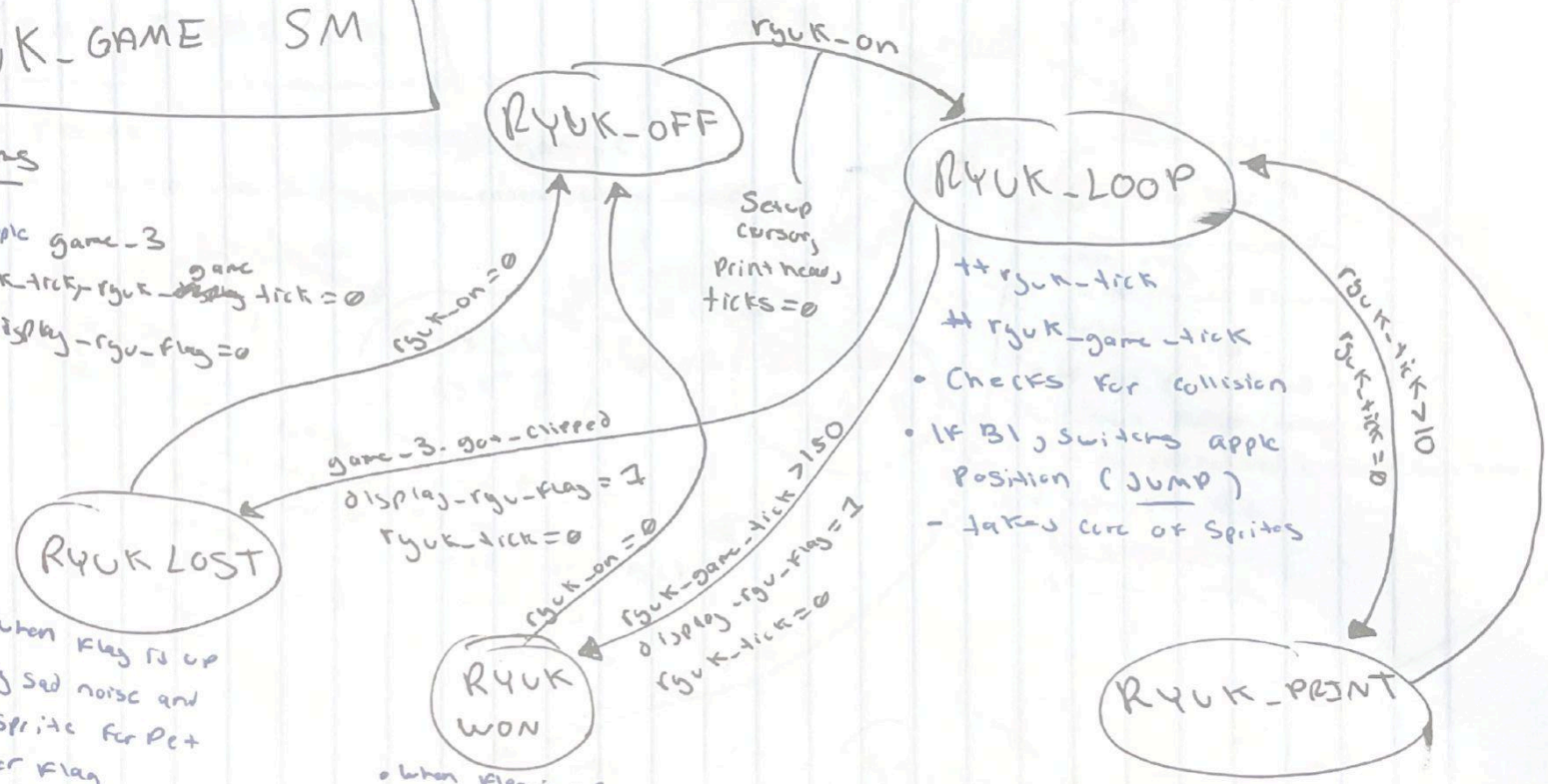
- HATCHes BOTH pets depending on certain factors.
- There are certain conditions to guarantee a pet.
  - if none are met, random pet

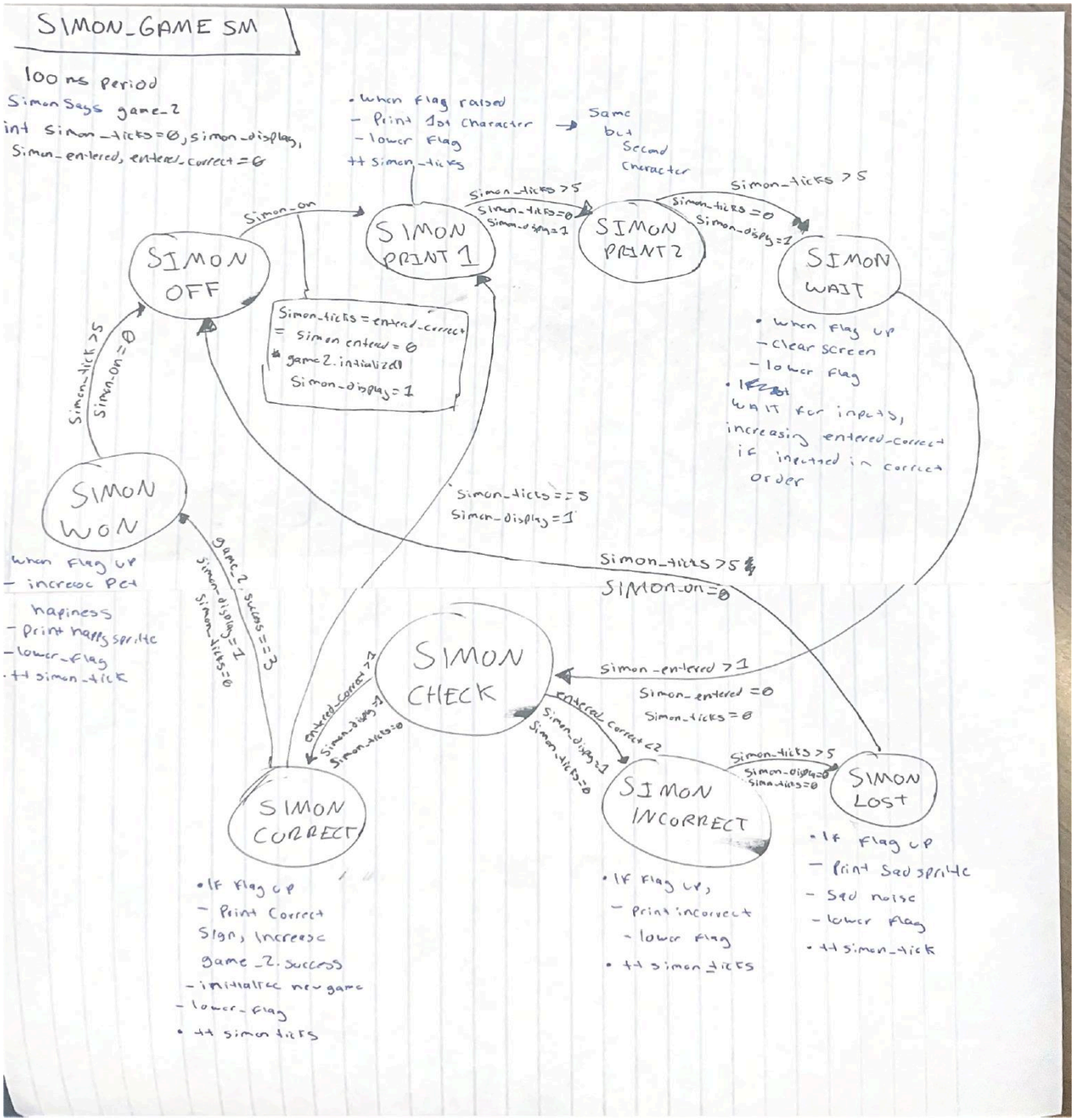
- Increases hunger and lowers happiness of both pets
- If levels are in bad thresholds, add a strike to Pet.strikes
  - if 3 strikes, pet dies and is reset to egg

RYUK_GAME SM

P: 100 ms

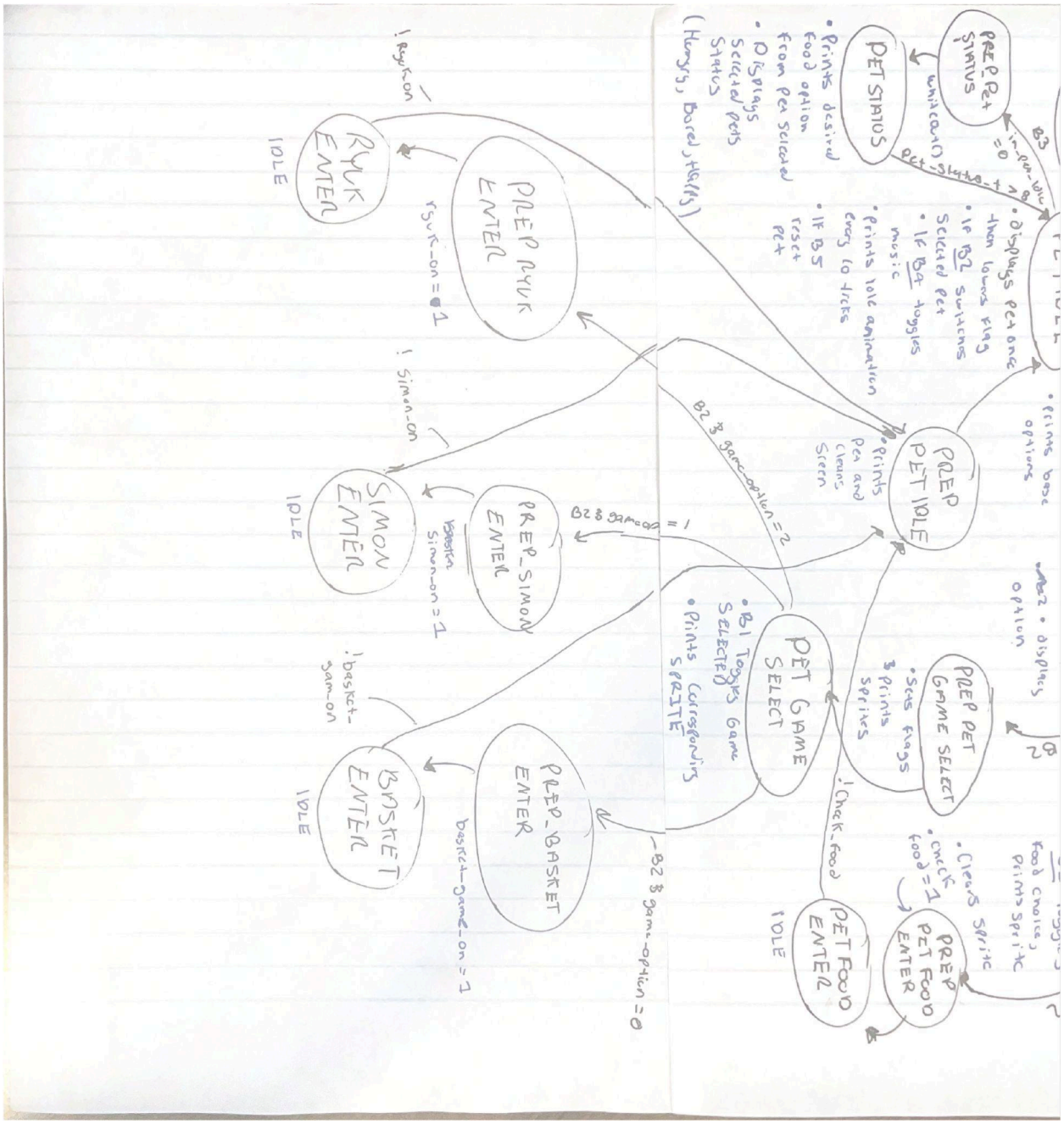
FlappyApple game_3
int ryuk_tick, ryuk_game_tick = 0
bool display_ryu_flag = 0




SIMON_GAME SM
100 ms period
Simon Says game_2
int simon_ticks=0, simon_display,
Simon_entered, entered_correct=0
• when flag raised
- Print 1st character
- lower flag
++ simon_ticks
Same but Second character
SIMON OFF
Simon_on
SIMON PRINT 1
Simon_ticks > 5
Simon_ticks=0
Simon_display=1
SIMON PRINT 2
Simon_ticks > 5
Simon_ticks=0
Simon_display=1
SIMON WAIT
• when flag up
- clear screen
- lower flag
• WAIT for inputs, increasing entered_correct if inputted in correct order
Simon_ticks = entered_correct
= Simon entered = 0
game2.initialized
Simon_display = 1
Simon_ticks > 5
Simon_on=0
SIMON WON
when flag up
- increase Pet hapiness
- Print happy sprite
- lower_flag
• ++ simon_tick
Simon_ticks == 5
Simon_display = 1
Simon_ticks > 5
SIMON_on=0
game_2.success == 3
Simon_display=1
Simon_ticks=0
SIMON CHECK
Simon_entered > 1
Simon_entered = 0
Simon_ticks = 0
entered_correct > 1
Simon_display=1
Simon_ticks=0
entered_correct < 2
Simon_display=1
Simon_ticks=0
Simon_ticks > 5
Simon_display=0
Simon_ticks=0
SIMON CORRECT
• If flag up
- Print Correct Sign, Increase game_2.success
- initialize new game
- lower_flag
• ++ simon ticks
SIMON INCORRECT
• If flag up,
- Print incorrect
- lower flag
• ++ simon_ticks
SIMON LOST
• If flag up
- Print Sad sprite
- Sad noise
- lower flag
• ++ simon_tick

PET SM
100ms period
bool food_option=0, food=1, game=0, display select switch=1
int game_option=0, stats_tick=0, idle_tick=0
PET INIT
PET IDLE
• displays pet once then lowers flag
• If B2 switches selected pet
• If B4 toggles music
• Prints idle animation every 10 ticks
• If B5 reset pet
B1 in pet idle =0
PET PREP SELECT
• Clears screen
• prints base options
PET_SELECT
• B1 toggles selected option
• B2 • displays option
food & B2
game & B2
PREP PET_FOOD SELECT
• Sets sprites and proper flag values
PET FOOD SELECT
• B1 toggles food choice, prints sprite
B2
• Clears sprite
• check food = 1
PREP PET FOOD ENTER
PET FOOD ENTER
IDLE
!Check_food
PREP PET GAME SELECT
• Sets flags & prints sprites
PET GAME SELECT
• B1 Toggles Game SELECTED
• Prints Corresponding SPRITE
B2 & game_option = 2
B2 & game_option = 1
B2 & game_option = 0
PREP PET IDLE
• Prints pet and clears screen
B3 in_pet_idle =0
PREP_Pet STATUS
white(out)
Pet_status_t > 8
PET STATUS
• Prints desired food option from pet selected
• Displays selected pets status
(Hungry, Bored, Happy)
PREP RYUK ENTER
ryuk_on = 1
RYUK ENTER
IDLE
! Ryuk_on
PREP_SIMON ENTER
simon_on = 1
SIMON ENTER
IDLE
! simon_on
PREP_BASKET ENTER
basket_game_on = 1
BASKET ENTER
IDLE
! basket_gam_on

PET SM
100ms period
bool food_option=0, food=1, game=0, display select switch=1
int game_option=0, status_tick=0, idle_tick=0
PET INIT
PET IDLE
• displays pet once then lowers flag
• If B2 switches selected pet
• If B4 toggles music
• Prints idle animation every 10 ticks
• If B5 reset pet
B1 in pet idle =0
PET PREP SELECT
• Clears screen
• prints base options
PET_SELECT
• B1 toggles selected option
• B2 • displays option
food & B2
game & B2
PREP PET_FOOD SELECT
• Sets sprites and proper flag values
PET FOOD SELECT
• B1 toggles food choice, prints sprite
B2
• Clears sprite
• check food = 1
PREP PET FOOD ENTER
PET FOOD ENTER
IDLE
!Check_food
PREP PET GAME SELECT
• Sets flags & prints sprites
PET GAME SELECT
• B1 Toggles Game SELECTED
• Prints Corresponding SPRITE
B2 & game_option = 0
PREP_BASKET ENTER
basket_game_on = 1
B2 & game_option = 1
PREP_SIMON ENTER
simon_on = 1
B2 & game_option = 2
PREP RYUK ENTER
ryuk_on = 1
PREP PET IDLE
• Prints pet and clears screen
B3 in_pet_idle =0
PREP_Pet STATUS
whiteout()
PET STATUS
• Prints desired food option from pet selected
• Displays selected pets status
(Hungry, Bored, Happy)
pet_status_t > 8


PREP PET STATUS
PET STATUS
PREP PET IDLE
PET GAME SELECT
PREP PET GAME SELECT
PET FOOD ENTER
PREP PET FOOD ENTER
PREP RYUK ENTER
RYUK ENTER
PREP_SIMON ENTER
SIMON ENTER
PREP_BASKET ENTER
BASKET ENTER
IDLE

Resources: