EMBEDDED SYSTEM DESIGN PROJECT 4: MULTITHREADED VIDEO GAME (V0.94)

OVERVIEW

For this project you will create a video game with sound effects. ECE 561 students will also add code to measure and display the system's time and memory use.

ECE 461 AND 561

REQUIREMENTS

Use the starter code provided for project 4. This includes the RTX real-time kernel.

Your code should use these threads/ISRs to interface with hardware. You may wish to modify them.

- Inputs
  - Task_Read_Accelerometer: Read the accelerometer at least every 100 ms (TBD) to determine board orientation. Use a mailbox to send the accelerometer information to your game thread.
  - Task_Read_TS: Read the touchscreen to determine additional user input. Use a mailbox to send the touch position information to your game thread.
- Outputs
  - Task_Sound: Writes a single sample of audio data to the digital to analog converter (DAC0). Its output is amplified to drive a speaker. Note: this code will likely be upgraded to improve performance.

In order to communicate with each other, processing threads (including ISRs) must use a kernel communication mechanism such as mutex, semaphore, event, mailbox, etc. Be sure to protect shared resources (e.g. LCD) using mutexes when needed. Any global variables used must be justified in your report.

STARTER CODE DOCUMENTATION

Full documentation for the starter code is posted online. The code and documentation will be updated as needed. Here is summary information on the API to help you in planning your software.

```
#define TFT_WIDTH (240)
#define TFT_HEIGHT (320)
typedef struct {
   uint32_t X, Y;
} PT_T;
typedef struct {
   uint8_t R, G, B; // note: using 5-6-5 color mode for LCD.
 // Values are left aligned here
} COLOR_T;

extern void TFT_Init(void);
extern void TFT_Fill_Buffer(COLOR_T * color);
extern void TFT_Plot_Pixel(PT_T * pos, COLOR_T * color);
extern void DrawLine(PT_T * p1, PT_T * p2, COLOR_T * color);
extern void TFT_Set_Backlight_Brightness(uint16_t brightness_percent);
extern void TFT_Set_Colors(COLOR_T * foreground, COLOR_T * background);

#define ROW_TO_Y(r) ((r)*G_TFT_char_height)
#define COL_TO_X(c) ((c)*G_TFT_char_width)
#define TFT_MAX_COLS (TFT_WIDTH/G_TFT_char_width)
#define TFT_MAX_ROWS (TFT_HEIGHT/G_TFT_char_height)

extern void TFT_Erase(void);
extern void TFT_Text_Init(uint8_t font_num);
extern void TFT_Text_PrintChar(PT_T * pos, char ch);
extern void TFT_Text_PrintStr(PT_T * pos, char * str);
extern void TFT_Text_PrintStr_RC(uint8_t row, uint8_t col, char * str);

extern void TFT_TS_Init(void);
extern uint32_t TFT_TS_Read(PT_T * position);
extern void TFT_TS_Test(void);
extern void TFT_TS_Calibrate(void);
```

ECE 561

Display system status on the screen when selected (e.g. by pressing on a "Status" label on the screen). Display the following information. You can choose whether to display it numerically, graphically, or both ways.

- CPU Time Utilization
  - CPU utilization per task
  - Total CPU utilization
  - CPU Idle time (time spend executing the idle task)
- Stack Memory Use
  - Current and maximum stack depth use per task

EXTRA CREDIT OPPORTUNITIES

- Load audio data from a microSD card.
- Display current worst-case response times (from release to task completion) per task on the status screen.

DELIVERABLES

- Milestone 1
    - Software design
        - Architecture / Structure
            - Per-thread information
                - Triggering information
                - Top-level design (flow chart or state chart, based on behavior). See example in appendix.
                - Design for any complex or critical processing.
            - Inter-thread communication
                - Events
                - Mutual exclusion
                - Global data
        - Behavior
            - Sequence diagrams with hardware and software actors
- Milestone 2 (final)
    - Final software design document
    - Development effort (time) tracking
        - Estimated person-hours required
        - Actual person-hours spent
    - Source code.
    - Video demonstrating game

GRADING

Your grade will depend on these factors:

- Functionality
- Correct use of RTOS mechanisms
- Game “wow” factor
- Quality of report

APPENDIX

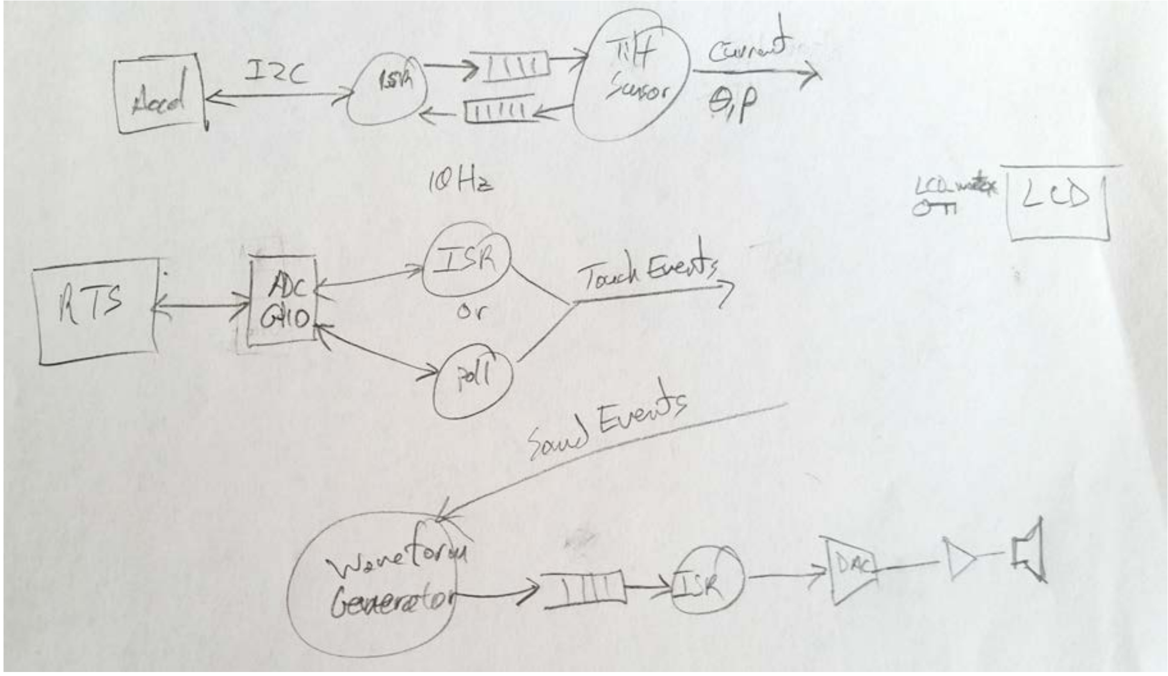

Figure 1. Example of communication between tasks which interface with hardware.

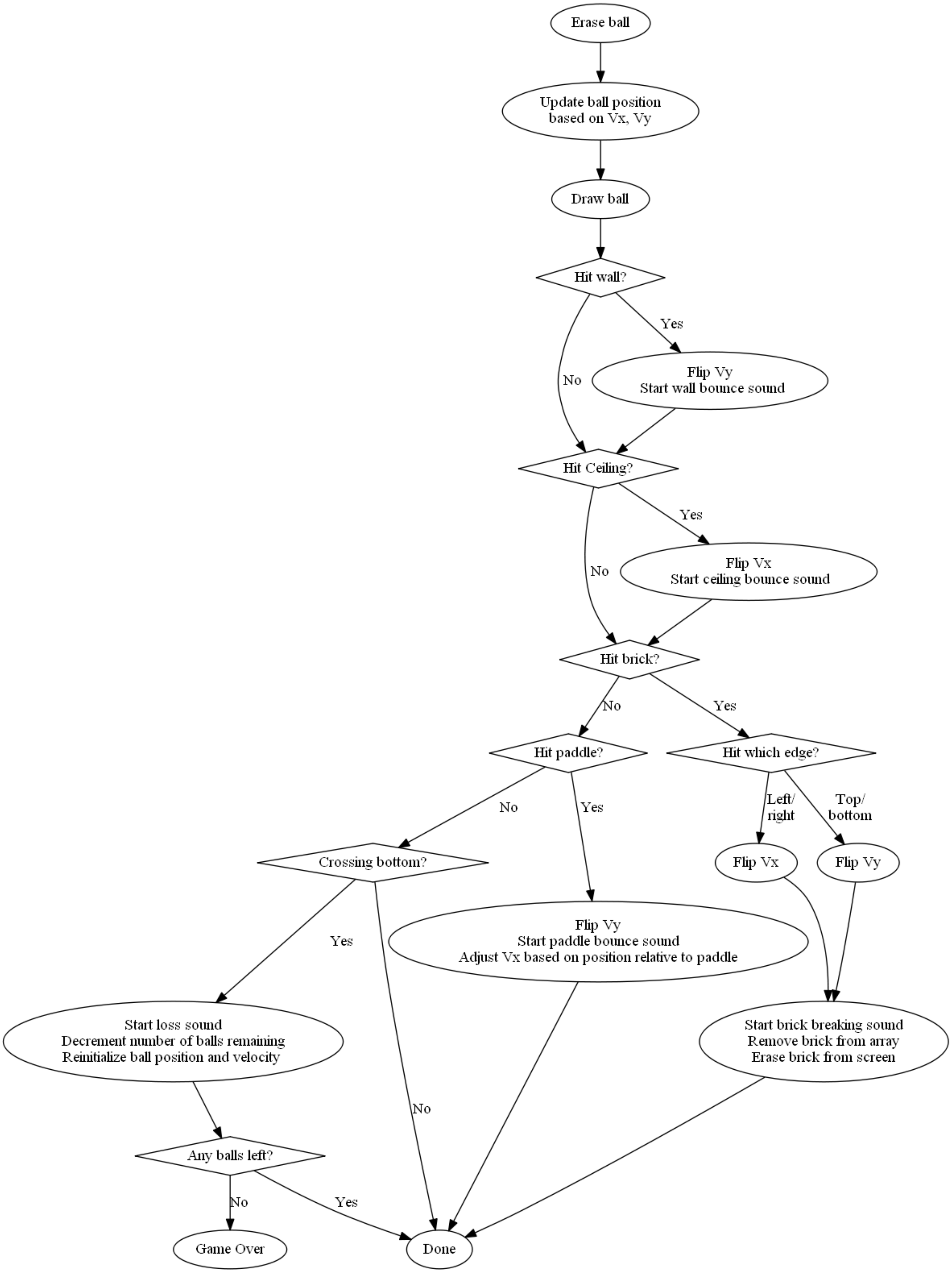
Erase ball
Update ball position based on Vx, Vy
Draw ball
Hit wall?
Yes
No
Flip Vy Start wall bounce sound
Hit Ceiling?
Yes
No
Flip Vx Start ceiling bounce sound
Hit brick?
No
Yes
Hit paddle?
Hit which edge?
No
Yes
Left/ right
Top/ bottom
Crossing bottom?
Flip Vx
Flip Vy
Flip Vy Start paddle bounce sound Adjust Vx based on position relative to paddle
Yes
Start loss sound Decrement number of balls remaining Reinitialize ball position and velocity
Start brick breaking sound Remove brick from array Erase brick from screen
No
Any balls left?
No
Yes
Game Over
Done

Figure 2. Example flow chart for Breakout game thread