Báo cáo bài tập cuối kì

## Thông tin tác giả

* Họ và tên: Trần Đức Nam Anh
* Mã số sinh viên: 20235655

## Bài tập 1

### Tóm tăt đề bài

*Xây dựng máy tính bỏ túi sử dụng RISC-V Assembly trên nền RARS/Digital Lab Sim, sử dụng 2 ngoại vi: keypad 4x4 và LED 7 đoạn để nhập, tính toán và hiển thị kết quả của các phép toán +, -, \*, /, % với các số nguyên.*

### Nguyên lý hoạt động chính

* *Khi nhấn phím số, số nhập sẽ được lưu dồn dần và hiển thị trên LED 7 đoạn. Do chỉ có 2 LED nên chỉ hiển thị 2 chữ số cuối cùng.*
* *Sau khi nhập số, có thể nhập toán tử +, -, \*, /, % bằng cách nhấn các phím tương ứng a, b, c, d, e.*
* *Khi nhấn phím f (=), chương trình sẽ tính toán và hiển thị kết quả trên LED.*
* *Chương trình hỗ trợ tính toán liên tiếp và tính năng repeat operation (ấn dấu = nhiều lần để lặp phép tính cuối)*

### Chi tiết về chương trình

*Mô tả về tác dụng của các hàm chính, tham số vào ra của hàm là gì. (Gợi ý nên chú thích trực tiếp vào trong mã nguồn rồi copy phần chú thích đầu hàm vào báo cáo)*

*# @brief Xử lý phím số, cập nhật giá trị nhập và hiển thị lên LED.*

*# @param[in] s1 Giá trị số đang nhập.*

*# @param[in] t4 Số vừa nhấn.*

*# @return Không có giá trị trả về, cập nhật số mới và hiển thị.*

*# @note Khi s5 = 1 (vừa nhấn =), sẽ reset toàn bộ biến trước khi nhập số mới.*

***Hàm capnhat***

*# @brief Xử lý phím phép toán (+, -, \*, /, %).*

*# @param[in] s1 Giá trị số đang nhập.*

*# @param[in] a4 Phép toán trước đó.*

*# @return Cập nhật kết quả trung gian nếu đã có phép toán trước đó.*

*# @note Reset cờ repeat khi nhập phép toán mới.*

***Hàm phep\_tinh***

*# @brief Xử lý phím phép toán (+, -, \*, /, %).*

*# @param[in] s1 Giá trị số đang nhập.*

*# @param[in] a4 Phép toán trước đó.*

*# @return Cập nhật kết quả trung gian nếu đã có phép toán trước đó.*

*# @note Reset cờ repeat khi nhập phép toán mới.*

***Hàm phep\_bang***

*# @brief Thực hiện phép tính lặp lại khi người dùng nhấn '=' nhiều lần.*

*# @param[in] s0, s6, s7 Kết quả và toán tử repeat.*

*# @return Hiển thị kết quả phép tính lặp.*

***Hàm repeat\_operation***

### Các giới hạn, ràng buộc nhập liệu đã thực hiện

* *Chỉ nhập tối đa 2 chữ số trên LED (giới hạn hiển thị).*
* *Khi chia cho 0, chương trình sẽ hiển thị thông báo lỗi và reset kết quả.*
* *Chương trình tự động reset giá trị khi bắt đầu nhập số mới sau khi nhấn '='.*
* *Hỗ trợ repeat operation: khi ấn '=' nhiều lần, lặp phép tính cuối với cùng toán hạng.*

### Hiểu về kiến trúc máy tính

 **Tổng số lượng lệnh máy:** khoảng 300+ lệnh.

 **Dung lượng bộ nhớ dữ liệu:** 64 bytes.

 **Dung lượng bộ nhớ chương trình:** khoảng 6 KB.

* Đếm số lượng lệnh theo 6 khuôn dạng:
  + R : ~100
  + I : ~120
  + J : ~30
  + S: ~20
  + U: ~10
  + B: ~30
* Số lượng cổng vào ra nếu có sử dụng:

 2 cổng ra LED 7 đoạn (SEVENSEG\_LEFT và SEVENSEG\_RIGHT)

 1 cổng vào bàn phím (IN\_ADDRESS\_HEXA\_KEYBOARD)

 1 cổng ra bàn phím (OUT\_ADDRESS\_HEXA\_KEYBOARD)

* Hãy chọn ra 2 biến và 2 label/nhãn được sử dụng trong bài và cho biết địa chỉ của chúng
  + Biến …A….. : địa chỉ 0x10010000…….
  + Biến …error….. : địa chỉ 0x1001000a…….
  + Nhãn …calc\_add.. : địa chỉ 0x0040021c
  + Nhãn …calc\_div….. : địa chỉ 0x00400234…….

*(Gợi ý: một số sai lệch nhỏ, do quá trình đếm hoặc do cập nhật chương trình mới là chấp nhận được)s*

## Bài tập 2

### Tóm tăt đề bài

*Xây dựng chương trình phát nhạc theo kịch bản bằng RISC-V Assembly trên nền RARS/Digital Lab Sim. Sử dụng ngoại vi keypad để chọn bản nhạc cần phát và dừng phát. Chương trình cho phép phát nhạc theo từng nốt được lưu trữ sẵn dưới dạng tham số (cao độ, thời lượng, loại nhạc cụ, âm lượng).*

### Nguyên lý hoạt động chính

* *Người dùng nhấn phím 1 đến 4 để chọn bản nhạc cần phát.*
* *Khi nhấn phím 0, chương trình dừng phát nhạc.*
* *Mỗi bản nhạc là một danh sách các bộ tham số: cao độ, thời lượng, loại nhạc cụ, âm lượng.*
* *Chương trình sử dụng ngắt ngoài để phát hiện phím nhấn.*
* *Khi có ngắt keypad, handler xử lý phím và gán giá trị vào thanh ghi a0.*
* *Dựa trên a0, chương trình chọn bản nhạc phù hợp và phát từng nốt nhạc.*
* *Sử dụng syscall 31 để phát nhạc và syscall 32 để delay giữa các nốt.*

### Chi tiết về chương trình

*Mô tả về tác dụng của các hàm chính, tham số vào ra của hàm là gì. (Gợi ý nên chú thích trực tiếp vào trong mã nguồn rồi copy phần chú thích đầu hàm vào báo cáo)*

*# @brief Xử lý ngắt ngoài khi người dùng nhấn phím keypad.*

*# @param[in] a0 Giá trị phím nhấn từ keypad.*

*# @return Cập nhật a0 để main loop nhận giá trị phím và xử lý.*

*# @note Sử dụng csrrw và uepc để trở về main loop sau khi xử lý.*

***Hàm handler***

*# @brief Phát nhạc theo kịch bản từ mảng tham số.*

*# @param[in] t2 Con trỏ mảng tham số (music1, music2,...).*

*# @return Phát nhạc từng nốt theo thứ tự, dừng khi gặp -1.*

*# @note Gọi syscall 31 để phát nhạc và syscall 32 để delay giữa các nốt.*

***Hàm play\_music***

*# @brief Tạm dừng phát nhạc khi người dùng nhấn phím 0.*

*# @param[in] Không có tham số trực tiếp.*

*# @return Chờ người dùng chọn bản nhạc mới để phát tiếp.*

***Hàm pause\_loop***

*# @brief Thực hiện phép tính lặp lại khi người dùng nhấn '=' nhiều lần.*

*# @param[in] s0, s6, s7 Kết quả và toán tử repeat.*

*# @return Hiển thị kết quả phép tính lặp.*

***Hàm repeat\_operation***

### Các giới hạn, ràng buộc nhập liệu đã thực hiện

* *Bản nhạc được lưu trong bộ nhớ dưới dạng tham số, mỗi nốt chiếm 16 byte.*
* *Dừng phát nhạc khi gặp pitch = -1.*
* *Dừng phát nhạc khi người dùng nhấn phím 0.*
* *Không giới hạn độ dài bản nhạc (miễn là không vượt quá bộ nhớ cấp phát).*

### Hiểu về kiến trúc máy tính

 Tổng số lượng lệnh máy: khoảng 200+ lệnh.

 Dung lượng bộ nhớ dữ liệu: 10000 bytes (A, B).

 Dung lượng bộ nhớ chương trình: khoảng 5 KB.

* Đếm số lượng lệnh theo 6 khuôn dạng:
  + R : ~60
  + I : ~90
  + J : ~20
  + S: ~15
  + U: ~10
  + B: ~20
* Số lượng cổng vào ra nếu có sử dụng:

 1 cổng vào keypad (IN\_ADDRESS\_HEXA\_KEYBOARD)

 1 cổng ra keypad (OUT\_ADDRESS\_HEXA\_KEYBOARD)

* Hãy chọn ra 2 biến và 2 label/nhãn được sử dụng trong bài và cho biết địa chỉ của chúng
  + Biến …A….. : địa chỉ 0x10010000…….
  + Biến …error….. : địa chỉ 0x1001273c…….
  + Nhãn …handler.. : địa chỉ 0x004000e0
  + Nhãn …get\_key\_code….. : địa chỉ 0x004000f0…….

*(Gợi ý: một số sai lệch nhỏ, do quá trình đếm hoặc do cập nhật chương trình mới là chấp nhận được)s*