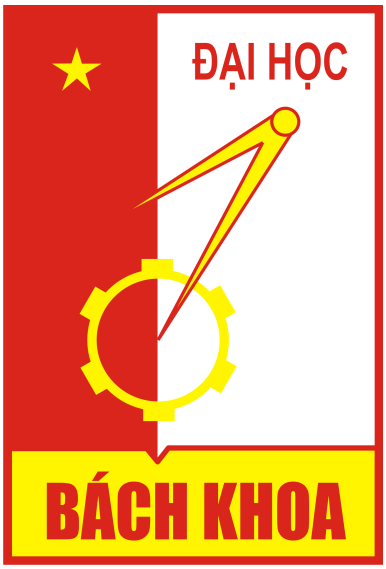
ĐẠI HỌC BÁCH KHOA HÀ NỘI

TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG



BÁO CÁO BÀI TẬP CUỐI KÌ

THỰC HÀNH KIẾN TRÚC MÁY TÍNH IT3280

Học kì 20232 - Năm học: 2023 - 2024

Giảng viên hướng dẫn: **TS. Hoàng Văn Hiệp**

Sinh viên: **Nguyễn Trung Kiên - 20226110**

Mã lớp: **147794**

Hà Nội, 2024

Mục lục

1 MÁY TÍNH BỎ TÚI

1.1 Đề bài

10. Máy tính bỏ túi

Sử dụng 2 ngoại vi là bàn phím và led 7 thanh để xây dựng một máy tính bỏ túi đơn giản. Hỗ trợ các phép toán +, -, *, /. Do trên bàn phím không có các phím trên nên sẽ dùng các phím

- Bấm phím a để nhập phép tính +
- Bấm phím b để nhập phép tính –
- Bấm phím c để nhập phép tính *
- Bấm phím d để nhập phép tính /
- Bấm phím f để nhập phép =

Yêu cầu cụ thể như sau:

- Khi nhấn các phím số, hiển thị lên LED, do chỉ có 2 LED nên chỉ hiện thị 2 số cuối cùng. Ví dụ khi nhấn phím 1 → hiện thị 01. Khi nhấn thêm phím 2 → hiển thị 12. Khi nhấn thêm phím 3 → hiển thị 23.
- Sau khi nhập số, sẽ nhập phép tính + - * /
- Sau khi nhấn phím f (dấu =) , tính toán và hiển thị kết quả lên LED.

Chú ý: Do bài toán sẽ có rất nhiều trường hợp xảy ra, yêu cầu cơ bản là thực hiện được phép tính và hiển thị lên LED. Các yêu cầu về bắt lỗi, các trường hợp tràn số, ... là mở rộng, không bắt buộc.

1.2 Định hướng cách làm

1.2.1 Ý tưởng

- Chương trình này mô phỏng một máy tính đơn giản, cho phép người dùng nhập các số và toán tử từ bàn phím hex, và hiển thị kết quả trên màn hình LED 7 thanh. Nó bao gồm các chức năng chính sau:
  - Kiểm tra bàn phím hex để nhận các phím được nhấn.
  - Xử lý các mã phím tương ứng với các số (0-9) và các toán tử (+, -, *, /, %, =).
  - Hiển thị các giá trị và kết quả trên LED 7 thanh.

1.3 Thuật toán và hàm

1.3.1 Khai báo dữ liệu

```
.eqv SEVENSEG_LEFT  0xFFFF0011
.eqv SEVENSEG_RIGHT 0xFFFF0010
.eqv IN_ADDRESS_HEXA_KEYBOARD       0xFFFF0012
.eqv OUT_ADDRESS_HEXA_KEYBOARD      0xFFFF0014
.eqv CODE_0                         0x11
.eqv CODE_1                         0x21
.eqv CODE_2                         0x41
.eqv CODE_3                         0x81
.eqv CODE_4                         0x12
.eqv CODE_5                         0x22
.eqv CODE_6                         0x42
.eqv CODE_7                         0x82
.eqv CODE_8                         0x14
.eqv CODE_9                         0x24
.eqv CODE_ADD                       0x44
.eqv CODE_SUB                       0x84
.eqv CODE_MUL                       0x18
.eqv CODE_DIV                       0x28
.eqv CODE_MOD                       0x48
```

```
.eqv CODE_EQL                       0x88
.data
NUMS_OF_7SEG:   .word       0x3F, 0x06, 0x5B, 0x4F, 0x66, 0x6D, 0x7D,
    0x07, 0x7F, 0x6F # Luu san dang ma LED 7 thanh cua so (0 -> 9) vao mang
str:    .asciiz "Ban nhap dau '=' khi chua nhap toan hang, hay thu lai \n "
.text
main:
    li      $t1,            IN_ADDRESS_HEXA_KEYBOARD
    li      $t2,            OUT_ADDRESS_HEXA_KEYBOARD
start:
    li      $s0,            0
    # Ma code cua phim duoc nhan.
    li      $s1,            0
    # Luu tru gia tri thuc cua phim duoc nhan (0 -> 15).
    li      $s2,            0
    # Che do hien tai cua chuong trinh. (1, 2, 3)
    li      $s3,            0
    # Toan hang cua phep tinh.
    li      $s4,            0
    # Toan tu.
    li      $s5,            0
    # Luu tru ket qua cua phep tinh truoc do.
    li      $s6,            0
    # Trang thai kiem tra toan hang da duoc nhap (0: chua nhap, 1: da nhap)
```

Các dữ liệu bao gồm:

- String thông báo tới người dùng.
- Địa chỉ của các cổng vào/ra (I/O ports) của hai LED 7 thanh: SEVENSEG_LEFT và SEVENSEG_RIGHT
- Địa chỉ các mã hiển thị LED 7 thanh của các chữ số và các toán tử: CODE_0 -> CODE_MOD
- Địa chỉ vào/ra của Hexa keyboard: IN_ADDRESS_HEXA_KEYBOARD và OUT_ADDRESS_HEXA_KEYBOARD
- Các biến được khai báo để lưu trữ thông tin chương trình. $s0 -> $s6

1.3.2 Chương trình polling để kiểm tra ký tự nhập vào

```

polling:
check_row_1:
    li      $t3,            0x01
    # Check 0, 1, 2, 3 (hang 1 cua ban phim)
    sb      $t3,            0($t1)
    # Luu gia tri hang can kiem tra vao $t1
    lbu     $a0,            0($t2)
    # Doc ma quet tu ban phim
    beq     $a0,            0,                          check_row_2

    # Neu khong co phim nao duoc nhan, chuyen sang kiem tra hang tiep theo
    bne     $a0,            $s0,                        code_processing

    # Neu ma quet khac voi ma truoc do, cap nhat ma
    beq     $a0,            $s0,                        back_to_polling

    # Neu ma quet giong ma truoc do, quay lai vong lap kiem tra
check_row_2:
    li      $t3,            0x02
    # Check 4, 5, 6, 7
    sb      $t3,            0($t1)
    # Luu gia tri hang can kiem tra vao $t1
```

```
24      lbu     $a0,            0($t2)
25      # Doc ma quet tu ban phim
26      beq     $a0,            0,                          check_row_3
27      bne     $a0,            $s0,                        code_processing
28      beq     $a0,            $s0,                        back_to_polling
29  check_row_2:
30      li      $t3,            0x02
31      sb      $t3,            0($t1)
32      lbu     $a0,            0($t2)
33      beq     $a0,            0,                          check_row_3
34      bne     $a0,            $s0,                        code_processing
35      beq     $a0,            $s0,                        back_to_polling
36  check_row_3:
37      li      $t3,            0x04
38      sb      $t3,            0($t1)
39      lbu     $a0,            0($t2)
40      beq     $a0,            0,                          check_row_4
41      bne     $a0,            $s0,                        code_processing
42      beq     $a0,            $s0,                        back_to_polling
43  check_row_4:
44      li      $t3,            0x08
45      sb      $t3,            0($t1)
46      lbu     $a0,            0($t2)
47      beq     $a0,            0,                          code_processing
48      bne     $a0,            $s0,                        code_processing
49      beq     $a0,            $s0,                        back_to_polling
50  code_processing:
51      add     $s0,            $zero,                      $a0
52      beq     $s0,            0,                          back_to_polling
53      beq     $s0,            CODE_0,                     process_code_0
54      beq     $s0,            CODE_1,                     process_code_1
55      beq     $s0,            CODE_2,                     process_code_2
56      beq     $s0,            CODE_3,                     process_code_3
57      beq     $s0,            CODE_4,                     process_code_4
58      beq     $s0,            CODE_5,                     process_code_5
59      beq     $s0,            CODE_6,                     process_code_6
60      beq     $s0,            CODE_7,                     process_code_7
61      beq     $s0,            CODE_8,                     process_code_8
62      beq     $s0,            CODE_9,                     process_code_9
63      beq     $s0,            CODE_ADD,                   process_code_add
64      beq     $s0,            CODE_SUB,                   process_code_sub
65      beq     $s0,            CODE_MUL,                   process_code_mul
66      beq     $s0,            CODE_DIV,                   process_code_div
67      beq     $s0,            CODE_MOD,                   process_code_mod
68      beq     $s0,            CODE_EQL,                   process_code_eql
69  process_code_0:
70      li      $s1,            0
71      li      $s2,            1
72      li      $s6,            1   # Danh dau la toan hang da duoc nhap
73      j       after_processing_code
74  process_code_1:
75      li      $s1,            1
76      li      $s2,            1
77      li      $s6,            1   # Danh dau la toan hang da duoc nhap
78      j       after_processing_code
79  process_code_2:
80      li      $s1,            2
81      li      $s2,            1
82      li      $s6,            1   # Danh dau la toan hang da duoc nhap
83      j       after_processing_code
84  process_code_3:
85      li      $s1,            3
86      li      $s2,            1
87      li      $s6,            1   # Danh dau la toan hang da duoc nhap
88      j       after_processing_code
89  process_code_4:
```

```
    li      $s1,            4
    li      $s2,            1
    li      $s6,            1  # Danh dau la toan hang da duoc nhap
    j       after_processing_code
process_code_5:
    li      $s1,            5
    li      $s2,            1
    li      $s6,            1  # Danh dau la toan hang da duoc nhap
    j       after_processing_code
process_code_6:
    li      $s1,            6
    li      $s2,            1
    li      $s6,            1  # Danh dau la toan hang da duoc nhap
    j       after_processing_code
process_code_7:
    li      $s1,            7
    li      $s2,            1
    li      $s6,            1  # Danh dau la toan hang da duoc nhap
    j       after_processing_code
process_code_8:
    li      $s1,            8
    li      $s2,            1
    li      $s6,            1  # Danh dau la toan hang da duoc nhap
    j       after_processing_code
process_code_9:
    li      $s1,            9
    li      $s2,            1
    li      $s6,            1  # Danh dau la toan hang da duoc nhap
    j       after_processing_code
process_code_add:
    li      $s1,            10
    li      $s2,            2
    j       after_processing_code
process_code_sub:
    li      $s1,            11
    li      $s2,            2
    j       after_processing_code
process_code_mul:
    li      $s1,            12
    li      $s2,            2
    j       after_processing_code
process_code_div:
    li      $s1,            13
    li      $s2,            2
    j       after_processing_code
process_code_mod:
    li      $s1,            14
    li      $s2,            2
    j       after_processing_code
process_code_eql:
    li      $s1,            15
    li      $s2,            3
    j       after_processing_code

after_processing_code:
    beq     $s2,            1,                          case1
    beq     $s2,            2,                          case2
    beq     $s2,            3,                          case3
```

- Polling
  - Vòng lặp chính để kiểm tra lần lượt từng hàng của bàn phím hex.
  - Đọc mã quét từ bàn phím và xác định phím được nhấn.
  - Chuyển đến hàm xử lý mã phím nếu phát hiện phím được nhấn.

- Code Processing
  - Xử lý mã phím được nhấn và xác định hành động tiếp theo dựa trên chế độ hiện tại của chương trình.
  - Chuyển đến các hàm xử lý cụ thể cho các phím số (0-9) và các toán tử (+, -, *, /, %, =).
- Process Code 0 đến Process Code 9
  - Xử lý các phím số từ 0 đến 9.
  - Cập nhật toán hạng và chế độ hiện tại sang Case_1 (chế độ xử lý toán hạng)
- Process Code Add, Process Code Sub, Process Code Mul, Process Code Div, Process Code Mod, Process Code Eql
  - Xử lý các phím toán tử (+, -, *, /, %, =).
  - Cập nhật toán tử và chế độ hiện tại sang Case_2 (chế độ xử lý toán tử)
- After Processing Code
  - Phát hiện dấu =
  - Chuyển chế độ hiện tại sang Case_3 (chế độ xử lý phép toán và tính kết quả)

1.3.3 Chế độ 1: Cập nhật toán hạng

```
# Mode 1: Neu toan tu cu la "=", loai bo thong tin cu. Cap nhat {toan hang} moi, xuat ra so moi tren man hinh de ta thay {toan hang} moi.
case1:
    beq     $s4,            15,                         case1_1
    j       case1_2
case1_1:
    li      $s3,            0
                                                        # Reset
    li      $s4,            0
                                                        # Reset
    li      $s5,            0
                                                        # Reset
case1_2:
    mul     $s3,            $s3,                        10
                        # Tinh toan lai gia tri toan hang
    add     $s3,            $s3,                        $s1
    add     $a0,            $zero,                      $s1
                       # In ra so moi tren man hinh
    li      $v0,            1
    syscall
    add     $a0,            $zero,                      $s3
    jal     render
        # In {toan hang} SEVENSEG
    j       sleep
```

- Nếu toán tử cũ là dấu "=", ta đặt các thông tin cũ về 0.
- Cập nhật toán hạng mới, xuất ra số mới trên màn hình để ta thấy toán hạng mới.
- Cụ thể, ta in số vừa nhập ra màn hình.
- Tính toán giá trị toán hạng mới: Ta nhân toán hạng cũ với 10 và cộng với số mới nhập vào.
- Chuyển giá trị toán hạng mới tới hàm "render" để xuất thông tin ra màn hình
- Cuối cùng, ta cho sleep để tiếp tục chờ và quay lại hàm polling

1.3.4 Chế độ 2: Cập nhật toán tử

```
# Che do 2: Neu {toan tu} cu khong thay doi, khong lam gi ca.
# Nguoc lai, cap nhat {toan tu}, cap nhat {ket qua} = {toan hang}, dau ra {toan hang} cu, xoa {toan hang} cu.
case2:
    beq     $s4,            $s1,                        sleep
    add     $s4,            $zero,                      $s1
                        # cap nhat {toan tu}
    add     $s5,            $zero,                      $s3
                        # cap nhat {ket qua} = {toan hang}
    add     $a0,            $zero,                      $s3
                        # dau ra {toan hang} cu
    jal     render
        # In {toan hang} SEVENSEG
    beq     $s1,            10,                         print_add
    beq     $s1,            11,                         print_sub
    beq     $s1,            12,                         print_mul
    beq     $s1,            13,                         print_div
    beq     $s1,            14,                         print_mod
case2_2:
    li      $s3,            0
                                                    # Xoa {toan hang} cu
    j       sleep
print_add:
    li      $a0,            '+'
                                                # In toan tu tuong ung
    li      $v0,            11
    syscall
    j case2_2

print_sub:
    li      $a0,            '-'
                                                # In toan tu tuong ung
    li      $v0,            11
    syscall
    j case2_2

print_mul:
    li      $a0,            '*'
                                                # In toan tu tuong ung
    li      $v0,            11
    syscall
    j case2_2

print_div:
    li      $a0,            '/'
                                                # In toan tu tuong ung
    li      $v0,            11
    syscall
    j case2_2

print_mod:
    li      $a0,            '%'
                                                # In toan tu tuong ung
    li      $v0,            11
    syscall
    j case2_2
```

- Thực hiện chương trình khi ta nhận được toán tử.
- Nếu toán tử cũ không thay đổi, không làm gì cả.
- Nếu toán tử có sự thay đổi, cập nhật toán tử.

- Ta lưu trữ giá trị toán hạng đã nhập trước đó hoặc là kết quả phép tính trước đó vào $s5.
- Gọi hàm "render" để in toán hạng hiện tại lên LED.
- So sánh giá trị thanh ghi $s1 với các giá trị đã gán sẵn tương ứng với các toán tử (+, -, *, /, %), nhảy đến các hàm print để in dấu tương ứng lên RUN I/O.

1.3.5 Chế độ 3: Xử lý phép toán và tính kết quả

```
# Che do 3: Cap nhat {ket qua} = {ket qua} {toan tu} {toan hang}, cap nhat {toan tu}, cap nhat {toan hang} = {ket qua}, dau ra {ket qua} moi.
case3:
    beq     $s6,            0,                          error_no_operand
        # Neu chua nhap toan hang, bao loi
    beq     $s4,            10,                         compu_add
    beq     $s4,            11,                         compu_sub
    beq     $s4,            12,                         compu_mul
    beq     $s4,            13,                         compu_div
    beq     $s4,            14,                         compu_mod
    beq     $s4,            15,                         compu_eql
compu_add:
    add     $s5,            $s5,                        $s3
    j       after_compu
compu_sub:
    sub     $s5,            $s5,                        $s3
    j       after_compu
compu_mul:
    mul     $s5,            $s5,                        $s3
    j       after_compu
compu_div:
    div     $s5,            $s3
    mflo    $s5
    j       after_compu
compu_mod:
    div     $s5,            $s3
    mfhi    $s5
    j       after_compu
compu_eql:
    j       after_compu
after_compu:
    li      $s4,            15
                                                    # Update {toan tu} "="
    add     $s3,            $zero,                      $s5
                            # Update {toan hang} = {ket qua}
    li      $a0,            '='
                                                    # In dau bang
    li      $v0,            11
    syscall
    add     $a0,            $zero,                      $s5
                            # Output {ket qua}
    li      $v0,            1
    syscall
    jal     render
        # Output {ket qua} den SEVENSEG
    j       sleep
sleep:
    li      $a0,            100
        # Sleep 100ms
    li      $v0,            32
    syscall
back_to_polling:
    j       polling                     # Continue polling
```

- Đầu tiên, khi ta nhập số thì ta gán thanh ghi $s6 là 1 để đánh dấu đã nhập toán hạng.
- Nếu ta nhập toán tử hoặc dấu "=" ở đầu tiên, $s6 có giá trị là 0. Khi đó, ta sẽ kiểm tra. Nếu chưa có toán hạng nhập vào, nhảy đến nhãn error_no_operand để báo với người dùng chưa có toán tử nhập vào.
- Tiếp theo, ta kiểm tra toán tử nhập vào và nhảy đến nhãn compu_... chứa phép tính tương ứng.
- Các phép tính thao tác với $s3 và $s5.
- Sau khi tính toán, lưu kết quả vào $s5. In dấu "=" và kết quả ra màn hình; sau đó gọi hàm render để hiển thị kết quả lên LED 7 thanh.
- Cuối cùng ta cho sleep và tiếp tục polling.

```
    error_no_operand:
    la      $a0,            str     # Thong bao loi
    li      $v0,            4
    syscall
    j       sleep
```

- Hàm này để in xâu str chương trình: thông báo cho người dùng chưa có toán hạng.

1.3.6 Hàm render

```
    # ham render:
# Tham so $a0 so nguyen can hien thi
render:
render_store:
    add     $sp,    $sp,    -24             # Mo rong stack
    sw      $ra,    20($sp)                 # Luu dia chi tra ve
    sw      $s0,    16($sp)                 # Luu gia tri thanh ghi $s0
    sw      $a0,    12($sp)             # Luu gia tri tham so $a0 (so
        nguyen can hien thi)
    sw      $a1,    08($sp)             # Luu gia tri tham so $a1 (dia chi
        cua led 7 thanh)
    sw      $t0,    04($sp)             # Luu gia tri thanh ghi $t0
    sw      $t1,    00($sp)             # Luu gia tri thanh ghi $t1
render_do:
    li      $t0,    10                  # Load 10 vao thanh ghi $t0
    add     $t1,    $zero,  $a0         # Sao chep gia tri tham so $a0 vao
        thanh ghi $t1
    div     $t1,    $t0                 # Chia $t1 cho 10
    mfhi    $a0                         # Lay phan du cua phep chia, chua
        hang don vi
    li      $a1,    SEVENSEG_RIGHT      # Dat dia chi cua led 7 thanh ben
        phai vao $a1
    jal     show_digit                  # Goi ham show_digit de hien thi
        so hang don vi
    mflo    $t1                         # Lay phan thap phan cua phep
        chia, chua hang chuc
    div     $t1,    $t0                 # Chia phan thap phan cho 10
    mfhi    $a0                         # Lay phan du cua phep chia, chua
        hang chuc
    li      $a1,    SEVENSEG_LEFT       # Dat dia chi cua led 7 thanh ben
        trai vao $a1
    jal     show_digit                  # Goi ham show_digit de hien thi
        so hang chuc

render_load:
```

```
26      lw      $t1,            00($sp)
                                                # Load
27      lw      $t0,            04($sp)
                                                # Load
28      lw      $a1,            08($sp)
                                                # Load
29      lw      $a0,            12($sp)
                                                # Load
30      lw      $s0,            16($sp)
                                                # Load
31      lw      $ra,            20($sp)
                                                # Load
32      add     $sp,            $sp,        +24
                                            # Thu ngan xep
33      jr      $ra
```

- Hàm render dùng để hiển thị hình ảnh led 7 thanh của số nguyên được đưa vào.
- Luôn hiển thị chữ số hàng chục và hàng đơn vị.
- Vì trong hàm render này ta phải sử dụng các thanh ghi, nên ta sẽ load các địa chỉ chương trình đang lưu vào một stack của $sp. Thực hiện ở nhãn render_store.
- Sau khi thực hiện xong tác vụ, ta lấy lại các địa chỉ đã lưu ra khỏi stack, trả về các thanh ghi và đóng stack. Thực hiện ở nhãn render_load.
- Nhãn render_do thực hiện tác vụ chính của hàm render. Ta chia lấy phần dư số cần hiển thị cho 10 để lấy giá trị hàng đơn vị và truyền vào thanh ghi $a1. Sau đó gọi hàm show_digit để xử lý.
- Tương tự, ta tiếp tục chia lấy phần dư cho 10 để lấy giá trị hành chục và truyền vào thanh ghi $a1. Sau đó gọi hàm show_digit để xử lý.

1.3.7 Hàm show_digit

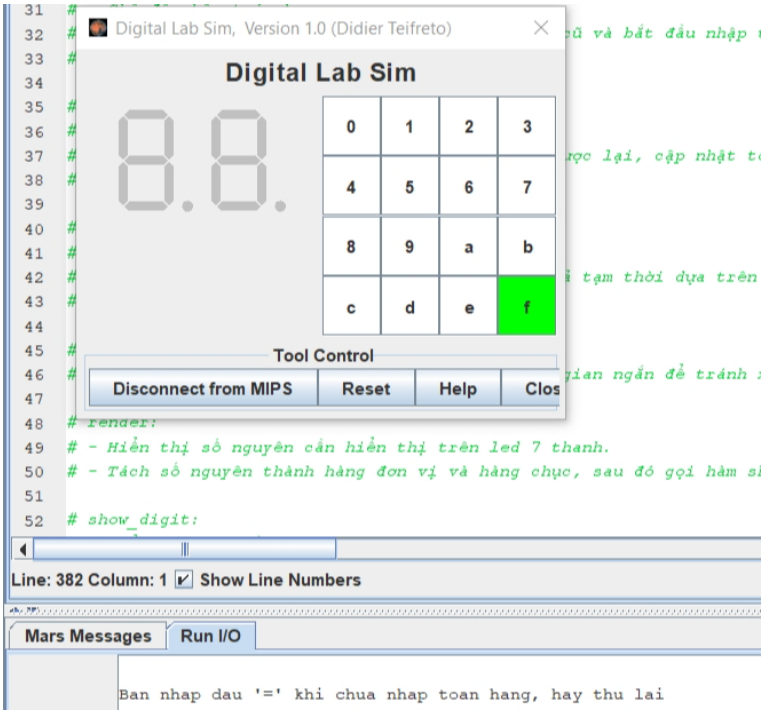
```
1       # Ham show_digit:
2   # Tham so $a0 la so can hien thi
3   # Tham so $a1 la ma SEVENSEG de hien thi
4
5   show_digit:
6   show_digit_store:
7       add     $sp,    $sp,    -12         # Mo rong stack
8       sw      $ra,    08($sp)             # Luu dia chi tra ve
9       sw      $t0,    04($sp)             # Luu gia tri thanh ghi $t0
10      sw      $t1,    00($sp)             # Luu gia tri thanh ghi $t1
11
12  show_digit_do:
13      la      $t0,    NUMS_OF_7SEG        # Load dia chi cua mang
            NUMS_OF_7SEG vao $t0
14      sll     $t1,    $a0,    2           # Nhan $a0 (so can hien thi) voi 4
            (do dich trai 2 bit)
15      add     $t0,    $t0,    $t1         # Tinh dia chi cua
            NUMS_OF_7SEG[$a0]
16      lw      $t0,    0($t0)              # Load gia tri tu
            NUMS_OF_7SEG[$a0] vao $t0
17      sb      $t0,    0($a1)              # Ghi gia tri nay vao dia chi cua
            led 7 thanh ($a1)
18
19  show_digit_load:
20      lw      $t1,    00($sp)             # Load gia tri thanh ghi $t1 tu
            stack
21      lw      $t0,    04($sp)             # Load gia tri thanh ghi $t0 tu
            stack
```

```
22      lw      $ra,    08($sp)             # Load dia chi tra ve tu stack
23      add     $sp,    $sp,    +12         # Thu hep stack
24      jr      $ra                         # Tra ve
```

- Ta cũng mở stack tương tự để lưu trữ giống như hàm render.
- Truy cập vào mảng NUMS_OF_7SEG đã khai báo và lấy ra giá trị mã hiển thị LED 7 thanh tương ứng với số cần hiển thị.
- Đưa mã vào địa chỉ của led 7 thanh để hiển thị đèn.

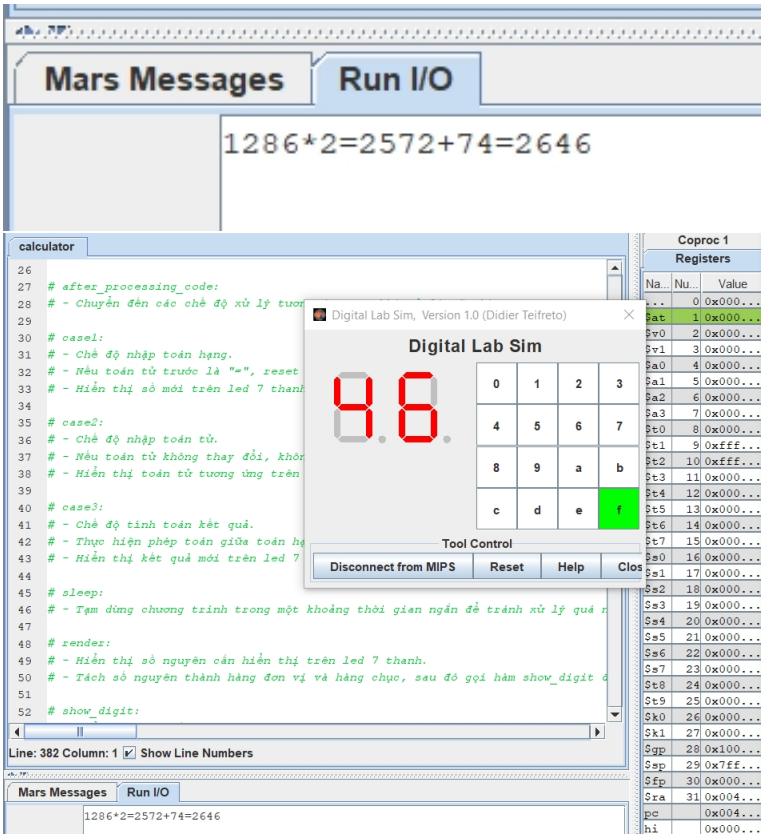
1.4 Kết quả

1.4.1 Khi nhập dấu "=" và không có toán hạng



1.5 Kết quả

1.5.1 Khi thực hiện liên tiếp các phép toán



→ Chương trình chạy đúng

1.6 SourceCode

Link clone Code của dự án: https://github.com/trungkienit25/IT3280_FinalProject.git

```
1
2  .eqv SEVENSEG_LEFT 0xFFFF0011
3  .eqv SEVENSEG_RIGHT 0xFFFF0010
4  .eqv IN_ADDRESS_HEXA_KEYBOARD 0xFFFF0012
5  .eqv OUT_ADDRESS_HEXA_KEYBOARD 0xFFFF0014
6  .eqv CODE_0 0x11
7  .eqv CODE_1 0x21
8  .eqv CODE_2 0x41
9  .eqv CODE_3 0x81
10 .eqv CODE_4 0x12
11 .eqv CODE_5 0x22
12 .eqv CODE_6 0x42
13 .eqv CODE_7 0x82
14 .eqv CODE_8 0x14
15 .eqv CODE_9 0x24
16 .eqv CODE_ADD 0x44
17 .eqv CODE_SUB 0x84
18 .eqv CODE_MUL 0x18
19 .eqv CODE_DIV 0x28
20 .eqv CODE_MOD 0x48
21 .eqv CODE_EQL 0x88
22
23 .data
24 NUMS_OF_7SEG: .word 0x3F, 0x06, 0x5B, 0x4F, 0x66, 0x6D, 0x7D, 0x07, 0x7F,
       0x6F # Luu san dang ma LED 7 thanh cua so (0 -> 9) vao mang
25 str: .asciiz "Ban nhap dau '=' khi chua nhap toan hang, hay thu lai \n"
```

```
26
27  .text
28  main:
29      li $t1, IN_ADDRESS_HEXA_KEYBOARD
30      li $t2, OUT_ADDRESS_HEXA_KEYBOARD
31  start:
32      li $s0, 0 # Ma code cua phim duoc nhan.
33      li $s1, 0 # Luu tru gia tri thuc cua phim duoc nhan (0 -> 15).
34      li $s2, 0 # Che do hien tai cua chuong trinh. (1, 2, 3)
35      li $s3, 0 # Toan hang cua phep tinh.
36      li $s4, 0 # Toan tu.
37      li $s5, 0 # Luu tru ket qua cua phep tinh truoc do.
38      li $s6, 0 # Trang thai kiem tra toan hang da duoc nhap (0: chua nhap,
            1: da nhap)
39
40  polling:
41  check_row_1:
42      li $t3, 0x01 # Check 0, 1, 2, 3 (hang 1 cua ban phim)
43      sb $t3, 0($t1) # Luu gia tri hang can kiem tra vao $t1
44      lbu $a0, 0($t2) # Doc ma quet tu ban phim
45      beq $a0, 0, check_row_2 # Neu khong co phim nao duoc nhan, chuyen sang
            kiem tra hang tiep theo
46      bne $a0, $s0, code_processing # Neu ma quet khac voi ma truoc do, cap
            nhat ma
47      beq $a0, $s0, back_to_polling # Neu ma quet giong ma truoc do, quay
            lai vong lap kiem tra
48
49  check_row_2:
50      li $t3, 0x02 # Check 4, 5, 6, 7
51      sb $t3, 0($t1) # Luu gia tri hang can kiem tra vao $t1
52      lbu $a0, 0($t2) # Doc ma quet tu ban phim
53      beq $a0, 0, check_row_3
54      bne $a0, $s0, code_processing
55      beq $a0, $s0, back_to_polling
56
57  check_row_3:
58      li $t3, 0x04 # Check 8, 9, a, b
59      sb $t3, 0($t1) # Luu gia tri hang can kiem tra vao $t1
60      lbu $a0, 0($t2) # Doc ma quet tu ban phim
61      beq $a0, 0, check_row_4
62      bne $a0, $s0, code_processing
63      beq $a0, $s0, back_to_polling
64
65  check_row_4:
66      li $t3, 0x08 # Check c, d, e, f
67      sb $t3, 0($t1) # Luu gia tri hang can kiem tra vao $t1
68      lbu $a0, 0($t2) # Doc ma quet tu ban phim
69      beq $a0, 0, code_processing
70      bne $a0, $s0, code_processing
71      beq $a0, $s0, back_to_polling
72
73  code_processing:
74      add $s0, $zero, $a0
75      beq $s0, 0, back_to_polling
76      beq $s0, CODE_0, process_code_0
77      beq $s0, CODE_1, process_code_1
78      beq $s0, CODE_2, process_code_2
79      beq $s0, CODE_3, process_code_3
80      beq $s0, CODE_4, process_code_4
81      beq $s0, CODE_5, process_code_5
82      beq $s0, CODE_6, process_code_6
83      beq $s0, CODE_7, process_code_7
84      beq $s0, CODE_8, process_code_8
85      beq $s0, CODE_9, process_code_9
86      beq $s0, CODE_ADD, process_code_add
87      beq $s0, CODE_SUB, process_code_sub
```

```
88      beq $s0, CODE_MUL, process_code_mul
89      beq $s0, CODE_DIV, process_code_div
90      beq $s0, CODE_MOD, process_code_mod
91      beq $s0, CODE_EQL, process_code_eql
92
93  process_code_0:
94      li $s1, 0
95      li $s2, 1
96      li $s6, 1 # Danh dau toan hang da duoc nhap
97      j after_processing_code
98
99  process_code_1:
100     li $s1, 1
101     li $s2, 1
102     li $s6, 1 # Danh dau toan hang da duoc nhap
103     j after_processing_code
104
105 process_code_2:
106     li $s1, 2
107     li $s2, 1
108     li $s6, 1 # Danh dau toan hang da duoc nhap
109     j after_processing_code
110
111 process_code_3:
112     li $s1, 3
113     li $s2, 1
114     li $s6, 1 # Danh dau toan hang da duoc nhap
115     j after_processing_code
116
117 process_code_4:
118     li $s1, 4
119     li $s2, 1
120     li $s6, 1 # Danh dau toan hang da duoc nhap
121     j after_processing_code
122
123 process_code_5:
124     li $s1, 5
125     li $s2, 1
126     li $s6, 1 # Danh dau toan hang da duoc nhap
127     j after_processing_code
128
129 process_code_6:
130     li $s1, 6
131     li $s2, 1
132     li $s6, 1 # Danh dau toan hang da duoc nhap
133     j after_processing_code
134
135 process_code_7:
136     li $s1, 7
137     li $s2, 1
138     li $s6, 1 # Danh dau toan hang da duoc nhap
139     j after_processing_code
140
141 process_code_8:
142     li $s1, 8
143     li $s2, 1
144     li $s6, 1 # Danh dau toan hang da duoc nhap
145     j after_processing_code
146
147 process_code_9:
148     li $s1, 9
149     li $s2, 1
150     li $s6, 1 # Danh dau toan hang da duoc nhap
151     j after_processing_code
152
153 process_code_add:
```

```
154     li $s1, 10
155     li $s2, 2
156     j after_processing_code
157 
158 process_code_sub:
159     li $s1, 11
160     li $s2, 2
161     j after_processing_code
162 
163 process_code_mul:
164     li $s1, 12
165     li $s2, 2
166     j after_processing_code
167 
168 process_code_div:
169     li $s1, 13
170     li $s2, 2
171     j after_processing_code
172 
173 process_code_mod:
174     li $s1, 14
175     li $s2, 2
176     j after_processing_code
177 
178 process_code_eql:
179     li $s1, 15
180     li $s2, 3
181     j after_processing_code
182 
183 after_processing_code:
184     beq $s2, 1, case1
185     beq $s2, 2, case2
186     beq $s2, 3, case3
187 
188 # Mode 1: Neu toan tu cu la "=", loai bo thong tin cu. Cap nhat {toan
    hang} moi, xuat ra so moi tren man hinh de ta thay {toan hang} moi.
189 case1:
190     beq $s4, 15, case1_1
191     j case1_2
192 case1_1:
193     li $s3, 0 # Reset
194     li $s4, 0 # Reset
195     li $s5, 0 # Reset
196 case1_2:
197     mul $s3, $s3, 10 # Tinh toan lai gia tri toan hang
198     add $s3, $s3, $s1
199     add $a0, $zero, $s1 # In ra so moi tren man hinh
200     li $v0, 1
201     syscall
202     add $a0, $zero, $s3
203     jal render # In {toan hang} SEVENSEG
204     j sleep
205 
206 # Che do 2: Neu {toan tu} cu khong thay doi, khong lam gi ca.
207 # Nguoc lai, cap nhat {toan tu}, cap nhat {ket qua} = {toan hang}, dau ra
    {toan hang} cu, xoa {toan hang} cu.
208 case2:
209     beq $s4, $s1, sleep
210     add $s4, $zero, $s1 # cap nhat {toan tu}
211     add $s5, $zero, $s3 # cap nhat {ket qua} = {toan hang}
212     add $a0, $zero, $s3 # dau ra {toan hang} cu
213     jal render # In {toan hang} SEVENSEG
214     beq $s1, 10, print_add
215     beq $s1, 11, print_sub
216     beq $s1, 12, print_mul
217     beq $s1, 13, print_div
```

```
218     beq $s1, 14, print_mod
219 
220 case2_2:
221     li $s3, 0 # Xoa {toan hang} cu
222     j sleep
223 
224 print_add:
225     li $a0, '+' # In toan tu tuong ung
226     li $v0, 11
227     syscall
228     j case2_2
229 
230 print_sub:
231     li $a0, '-' # In toan tu tuong ung
232     li $v0, 11
233     syscall
234     j case2_2
235 
236 print_mul:
237     li $a0, '*' # In toan tu tuong ung
238     li $v0, 11
239     syscall
240     j case2_2
241 
242 print_div:
243     li $a0, '/' # In toan tu tuong ung
244     li $v0, 11
245     syscall
246     j case2_2
247 
248 print_mod:
249     li $a0, '%' # In toan tu tuong ung
250     li $v0, 11
251     syscall
252     j case2_2
253 
254 # Che do 3: Cap nhat {ket qua} = {ket qua} {toan tu} {toan hang}, cap nhat
      {toan tu}, cap nhat {toan hang} = {ket qua}, dau ra {ket qua} moi.
255 case3:
256     beq $s6, 0, error_no_operand # Neu chua nhap toan hang, bao loi
257     beq $s4, 10, compu_add
258     beq $s4, 11, compu_sub
259     beq $s4, 12, compu_mul
260     beq $s4, 13, compu_div
261     beq $s4, 14, compu_mod
262     beq $s4, 15, compu_eql
263 
264 compu_add:
265     add $s5, $s5, $s3
266     j after_compu
267 
268 compu_sub:
269     sub $s5, $s5, $s3
270     j after_compu
271 
272 compu_mul:
273     mul $s5, $s5, $s3
274     j after_compu
275 
276 compu_div:
277     div $s5, $s3
278     mflo $s5
279     j after_compu
280 
281 compu_mod:
282     div $s5, $s3
```

```
283     mfhi $s5
284     j after_compu
285
286 compu_eql:
287     j after_compu
288
289 after_compu:
290     li $s4, 15 # Update {toan tu} = "="
291     add $s3, $zero, $s5 # Update {toan hang} = {ket qua}
292     li $a0, '=' # In dau bang
293     li $v0, 11
294     syscall
295     add $a0, $zero, $s5 # Output {ket qua}
296     li $v0, 1
297     syscall
298     jal render # Output {ket qua} den SEVENSEG
299     j sleep
300
301 sleep:
302     li $a0, 100 # Sleep 100ms
303     li $v0, 32
304     syscall
305
306 back_to_polling:
307     j polling # Continue polling
308
309 # ham render:
310 # Tham so $a0 so nguyen can hien thi
311 render:
312 render_store:
313     add     $sp,    $sp,    -24             # Mo rong stack
314     sw      $ra,    20($sp)                 # Luu dia chi tra ve
315     sw      $s0,    16($sp)                 # Luu gia tri thanh ghi $s0
316     sw      $a0,    12($sp)             # Luu gia tri tham so $a0 (so
            nguyen can hien thi)
317     sw      $a1,    08($sp)             # Luu gia tri tham so $a1 (dia chi
            cua led 7 thanh)
318     sw      $t0,    04($sp)             # Luu gia tri thanh ghi $t0
319     sw      $t1,    00($sp)             # Luu gia tri thanh ghi $t1
320 render_do:
321     li      $t0,    10                  # Load 10 vao thanh ghi $t0
322     add     $t1,    $zero,  $a0         # Sao chep gia tri tham so $a0 vao
            thanh ghi $t1
323     div     $t1,    $t0                 # Chia $t1 cho 10
324     mfhi    $a0                         # Lay phan du cua phep chia, chua
            hang don vi
325     li      $a1,    SEVENSEG_RIGHT      # Dat dia chi cua led 7 thanh ben
            phai vao $a1
326     jal     show_digit                  # Goi ham show_digit de hien thi
            so hang don vi
327     mflo    $t1                         # Lay phan thap phan cua phep
            chia, chua hang chuc
328     div     $t1,    $t0                 # Chia phan thap phan cho 10
329     mfhi    $a0                         # Lay phan du cua phep chia, chua
            hang chuc
330     li      $a1,    SEVENSEG_LEFT       # Dat dia chi cua led 7 thanh ben
            trai vao $a1
331     jal     show_digit                  # Goi ham show_digit de hien thi
            so hang chuc
332
333 render_load:
334     lw      $t1,            00($sp)
                                            # Load
335     lw      $t0,            04($sp)
                                            # Load
```

```
336     lw      $a1,            08($sp)
                                            # Load
337     lw      $a0,            12($sp)
                                            # Load
338     lw      $s0,            16($sp)
                                            # Load
339     lw      $ra,            20($sp)
                                            # Load
340     add     $sp,            $sp,      +24
                                        # Thu ngan xep
341     jr      $ra
342     # Ham show_digit:
343 # Tham so $a0 la so can hien thi
344 # Tham so $a1 la ma SEVENSEG de hien thi
345
346 show_digit:
347 show_digit_store:
348     add     $sp,    $sp,    -12         # Mo rong stack
349     sw      $ra,    08($sp)             # Luu dia chi tra ve
350     sw      $t0,    04($sp)             # Luu gia tri thanh ghi $t0
351     sw      $t1,    00($sp)             # Luu gia tri thanh ghi $t1
352
353 show_digit_do:
354     la      $t0,    NUMS_OF_7SEG        # Load dia chi cua mang
            NUMS_OF_7SEG vao $t0
355     sll     $t1,    $a0,    2           # Nhan $a0 (so can hien thi) voi 4
            (do dich trai 2 bit)
356     add     $t0,    $t0,    $t1         # Tinh dia chi cua
            NUMS_OF_7SEG[$a0]
357     lw      $t0,    0($t0)              # Load gia tri tu
            NUMS_OF_7SEG[$a0] vao $t0
358     sb      $t0,    0($a1)              # Ghi gia tri nay vao dia chi cua
            led 7 thanh ($a1)
359
360 show_digit_load:
361     lw      $t1,    00($sp)             # Load gia tri thanh ghi $t1 tu
            stack
362     lw      $t0,    04($sp)             # Load gia tri thanh ghi $t0 tu
            stack
363     lw      $ra,    08($sp)             # Load dia chi tra ve tu stack
364     add     $sp,    $sp,    +12         # Thu hep stack
365     jr      $ra                         # Tra ve
366
367 error_no_operand:
368     la      $a0,            str     # Thong bao loi
369     li      $v0,            4
370     syscall
371     j       sleep
```