

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI  
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG



# BÁO CÁO CUỐI KỲ

## MÔN: THỰC HÀNH KIẾN TRÚC MÁY TÍNH

Giảng viên hướng dẫn: Thầy Lê Bá Vui

Nhóm sinh viên thực hiện:

1. Lê Đức Sơn - 20194658
2. Lương Thị Tâm - 20194663

Lớp: Thực hành kiến trúc máy tính - IT3280 - 130938

Hà Nội, tháng 7 năm 2022

# MỤC LỤC

|   |          |
|---|----------|
| <b>MỤC LỤC</b>  | <b>2</b> |
| <b>GIAO ĐỀ</b>  | <b>2</b> |
| <b>KẾT QUẢ THỰC HIỆN</b>                              | <b>3</b> |
| Bài 3: Kiểm tra tốc độ và độ chính xác khi gõ văn bản | 3        |
| Phân tích cách làm và thuật toán.                     | 3        |
| Mã nguồn  | 4        |
| Kết quả chạy mô phỏng                                 | 8        |
| Bài 5: Biểu thức trung tố hậu tố                      | 10       |
| Phân tích cách làm và thuật toán.                     | 10       |
| Mã nguồn  | 12       |
| Kết quả chạy mô phỏng                                 | 18       |

## **GIAO ĐỀ**

| SINH VIÊN THỰC HIỆN      | ĐỀ |
|--------------------------|----|
| Lê Đức Sơn - 20194658    | 3  |
| Lương Thị Tâm - 20194663 | 5  |

# KẾT QUẢ THỰC HIỆN

## Bài 3: Kiểm tra tốc độ và độ chính xác khi gõ văn bản

Thực hiện chương trình đo tốc độ gõ bàn phím và hiển thị kết quả bằng 2 đèn led 7 đoạn. Nguyên tắc:

- Cho một đoạn văn bản mẫu, cố định sẵn trong mã nguồn. Ví dụ “bo mon ky thuat may tinh”
- Sử dụng bộ định thời Timer (trong bộ giả lập Digital Lab Sim) để tạo ra khoảng thời gian để đo. Đây là thời gian giữa 2 lần ngắt, chu kỳ ngắt.
- Người dùng nhập các ký tự từ bàn phím. Ví dụ nhập “ bo mOn ky Shuat may tinh”. Chương trình cần phải đếm số ký tự đúng (trong ví dụ trên thì người dùng gõ sai chữ O và S) mà người dùng đã gõ và hiển thị lên các đèn led.
- Chương trình đồng thời cần tính được tốc độ gõ: thời gian hoàn thành và số từ trên một đơn vị thời gian.

### A. Phân tích cách làm và thuật toán.

- Sử dụng 1 vòng lặp vô hạn.  
Trong vòng lặp có kiểm tra giá trị tại địa chỉ KEY\_READY nếu bằng 1 thì thực hiện tạo ngắt bằng teqi  
Đồng thời chương trình cũng cho phép ngắt bằng bộ đếm time counter(timer)
- Khi đã bắt được exception và con trỏ \$pc nhảy đến vùng phục vụ ngắt .ktext  
Bên trong vùng .ktext ta sẽ lấy giá trị bên trong thanh ghi Coproc0.cause(\$13) để kiểm tra đây là loại ngắt nào
- Trong trường hợp lệnh ngắt được thực hiện bởi teqi(tạo ra khi nhận được ký tự từ bàn phím)  
Ta kiểm tra xem ký tự thứ i của string đã cho có phải là ký tự kết thúc hay không(“\0”), nếu đúng thì kết thúc chương trình, hiển thị ra số ký tự đúng lên Digital Lab Sim và in ra thời gian hoàn thành, tốc độ gõ lên màn hình.

Nếu chương trình chưa kết thúc, ta so sánh ký tự vừa nhập với ký tự string[i] nếu bằng nhau -> tăng biến đếm số ký tự đúng lên 1

Tiếp tục kiểm tra xem ký tự vừa nhập vào == ' ' && ký nhập vào trước đó(prv) != ' ' nếu đúng thì tăng biến đếm số từ đã nhập lên 1

Sau đó tăng số ký tự nhập vào trong 1s lên 1, cập nhật giá trị của prv bằng ký tự vừa nhập vào và chuyển con trỏ của string lên 1 để phục vụ cho lần sau

- Trong trường hợp lệnh ngắt được thực hiện bởi bộ đếm time counter(timer)

Kiểm tra xem số lần tạo lệnh ngắt của timer đã đủ chưa (1s), nếu chưa đủ thì tăng biến đếm lên, nếu đã đủ thì hiển thị số ký tự đã gõ trong 1s lên Digital Lab Sim và khởi tạo lại biến đếm ký tự trong 1s, đồng thời tăng biến đếm thời gian hoàn thành nhập lên 1s

- Sau khi hoàn thành các câu lệnh trong vùng .ktext

Thực hiện các hàm cần thiết để thiết lập lại các thông số để đón nhận lần ngắt tiếp theo

## B. Mã nguồn

```

1  #~~~~~
2  #          AUTHOR: LE DUC SON 20194658
3  #~~~~~
4  .eqv SEVENSEG_LEFT 0xFFFF0011 # Địa chỉ của đèn led 7 đoạn trái.
5  .eqv SEVENSEG_RIGHT 0xFFFF0010 # Địa chỉ của đèn led 7 đoạn phải
6  .eqv IN_ADDRESS_HEXKEYBOARD 0xFFFF0012
7  .eqv MASK_CAUSE_COUNTER 0x00000400 # Bit 10: Counter interrupt
8  .eqv COUNTER 0xFFFF0013 #Time Counter
9  .eqv KEY_CODE 0xFFFF0004 # ASCII code from keyboard, 1 byte
10 .eqv KEY_READY 0xFFFF0000 # =1 if has a new keycode ?
11 .data
12 num: .byte 63, 6, 91, 79, 102, 109, 125, 7, 127, 111
13 string: .ascii "Bỏ món kiến trúc máy tính"
14 mes1: .ascii "Thời gian hoàn thành: "
15 mes2: .ascii "(s) và tốc độ gõ trung bình: "
16 mes3: .ascii "tu/phút\n"
17 #~~~~~
18 #MAIN Procsciiz ciiz edure
19 #~~~~~
20 .text#global _v : k0, k1, s0, s1, s2, s3, s4, s5, a1
21 li $k0, KEY_CODE
22 li $k1, KEY_READY
23 li $t1, COUNTER #Khởi tạo bộ đếm timer
24 sb $t1, 0($t1)
25 addi $s0, $0, 0 #Đếm số ký tự trong 1s
26 addi $s1, $0, 0 #Đếm tổng số ký tự dùng
27 addi $s2, $0, 1 #Đếm tổng số ký tự nhập vào
28 addi $s3, $0, 0 #Đếm số lần counter_intr
29 addi $s4, $0, 0 #Lưu trữ ký tự trước đó
30 addi $s5, $0, 0 #Đếm thời gian (giây)

```

```

31     la      $a1, string
32     #~~~~~
33     #VONG LAP VO HAN DE DOI INTERRUPT
34 loop:
35     lw      $t1, 0($k1)          #$t1 = [$k1] = KEY_READY
36     bne     $t1, $zero, make_Keyboard_Intr  #Tao interrupt khi nhan duoc ky tu tu ban phim
37     addi    $v0, $0, 32
38     li      $a0, 5
39     syscall
40     b loop          #so lenh trong 1 vong lap = 6 => cu lap 5 lan thi tao 1 counter interrupt
41     nop
42     #~~~~~
43 make_Keyboard_Intr:
44     teqi     $t1, 1
45     b        loop          #quay lai vong lap de cho doi su kien interrupt tiep theo
46     nop
47 end_Main:
48     #~~~~~
49     #          PHAN PHUC VU NGAT
50     #~~~~~
51     ktext 0x80000180
52
53     dis_int:li $t1, COUNTER          # BUG: must disable with Time Counter
54     sb      $zero, 0($t1)
55     #~~~~~
56     #LAY GIA TRI CUA THANH GHI C0.cause DE KIEM TRA LOAI INTERRUPT
57     get_Caus:mfc0 $t1, $13          # $t1 = Coproc0.cause
58     isCount:li $t2, MASK_CAUSE_COUNTER# if Cause value confirm Counter..
59     and     $at, $t1,$t2
60     bne     $at,$t2, keyboard_Intr

```

```

61 #~~~~~
62 #NGAT DO BO DEM COUNTER
63 counter_Intr:
64     blt $s3, 40, continue      #Neu so lap ngat do counter = 40 : da du 1s
65     # -> khoi tao lai $s3, chieu toc do go ra DLS, tang bien dem thoi gian len 1
66     jal show
67     addi $s3, $0, 0            #Khoi tao lai $s3
68     addi $s5, $s5, 1          #Tang bien dem thoi gian(s)
69     j en_int
70     nop
71 continue:
72     addi $s3, $s3, 1          #Neu chua du 1s thi tang bien dem so lan ngat
73     j en_int
74     nop
75 keyboard_Intr:
76 #~~~~~
77 #NGAT DO BAN PHIM
78 check_Matching:              #Kiem tra ky tu nhap vao
79     lb $t0, 0($a1)            #Lay ki tu thu i trong mang da cho
80     beq $t0, $0, end_Program  #Dung chuong trinh neu gap ki tu '\0'
81     lb $t1, 0($k0)            #Lay ki tu nhap vao tu ban phim
82     beq $t1, $0, en_int       #bug
83     bne $t0, $t1, check_Space #Neu ki tu nhap vao va ki tu thu i trong mang da cho bang nhau
84     # -> $s1++(dem so ki tu dung)
85     nop
86     addi $s1, $s1, 1          #Tang bien dem so ky tu dung
87 check_Space:                  #Kiem tra ki tu nhap vao co phai la ' ' hay ko
88     bne $t1, ' ', end_Process #if(ky tu nhap vao == ' ' && string[i-1] != ' ') $s2++(dem so tu da nhap)
89     nop
90     beq $s4, ' ', end_Process

```

```

91     nop
92     addi $s2, $s2, 1           #Tang bien dem so tu da nhap
93 end_Process:
94     addi $s0, $s0, 1           #Tang so ky tu trong 1s len 1
95     addi $s4, $t1, 0           #Cap nhat lai thanh ghi chua ky tu nhap vao ban phim truoc do
96     addi $a1, $a1, 1           #Tang con tro len 1 <=> string+i
97     #~~~~~
98 en_int:
99     li $t1, COUNTER
100    sb $t1, 0($t1)
101    mtc0 $zero, $13 # Must clear cause reg
102 next_pc: mfc0 $at, $14 # $at <= Coproc0.$14 = Coproc0.epc
103    addi $at, $at, 4 # $at = $at + 4 (next instruction)
104    mtc0 $at, $14 # Coproc0.$14 = Coproc0.epc <= $at
105 return: eret # Return from exception
106    #~~~~~
107 # CHIEU RA MAN HINH DIGITAL LAB SIM GIA TRI CUA $s0
108    #~~~~~
109 show:
110    addi $sp, $sp, -4
111    sw $ra, ($sp)
112    addi $t0, $0, 10
113    div $s0, $t0
114    mflo $v1           #lay so hang chuc
115    mfhi $v0           #lay so hang don vi
116    la $a0, num
117    add $a0, $a0, $v1
118    lb $a0, 0($a0) # set value for segments
119    jal SHOW_7SEG_LEFT # show
120    la $a0, num

```

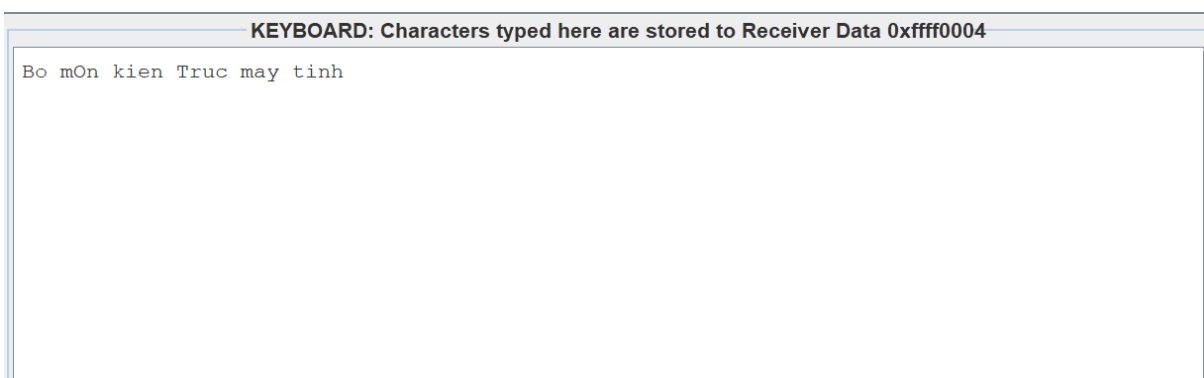


```


121     add $a0, $a0, $v0
122     lb $a0, 0($a0) # set value for segments
123     jal SHOW_7SEG_RIGHT # show
124     addi $s0, $0, 0 #Sau khi chieu ra man hinh thi khoi tao lai bien dem
125     lw $ra, ($sp)
126     addi $sp, $sp, 4
127     jr $ra
128 SHOW_7SEG_LEFT:
129     li $t0, SEVENSEG_LEFT # assign port's address
130     sb $a0, 0($t0) # assign new value
131     jr $ra
132 SHOW_7SEG_RIGHT:
133     li $t0, SEVENSEG_RIGHT # assign port's address
134     sb $a0, 0($t0) # assign new value
135     jr $ra
136     nop
137 #~~~~~
138 # KET THUC CHUONG TRINH VA HIEN THI SO KY TU DUNG
139 #~~~~~
140 end_Program:
141     addi $v0, $0, 4
142     la $a0, mes1
143     syscall
144     addi $v0, $0, 1
145     addi $a0, $s5, 0
146     syscall #In ra man hinh thoi gian hoan thanh
147     addi $v0, $0, 4
148     la $a0, mes2
149     syscall
150     addi $v0, $0, 1
151     addi $a0, $0, 60
152     mult $s2, $a0
153     mflo $s2
154     div $s2, $s5
155     mflo $a0
156     syscall #In ra man hinh toc do go trung binh
157     addi $v0, $0, 4
158     la $a0, mes3
159     syscall
160     addi $s0, $s1, 0
161     jal show #Chieu ra man hinh DLS so ky tu dung

```

## C. Kết quả chạy mô phỏng



## Digital Lab Sim



|   |   |   |   |
|---|---|---|---|
| 0 | 1 | 2 | 3 |
| 4 | 5 | 6 | 7 |
| 8 | 9 | a | b |
| c | d | e | f |

**Tool Control**

Disconnect from MIPS

Reset

Help

Close

Mars Messages

Run I/O

Clear

```
-- program is finished running (dropped off bottom) --
Thoi gian hoan thanh: 12(s) va toc do go trung binh: 30 tu/phut
-- program is finished running (dropped off bottom) --
```

## Bài 5: Biểu thức trung tố hậu tố

Viết chương trình tính giá trị biểu thức bất kỳ bằng phương pháp duyệt biểu thức hậu tố.

Các yêu cầu cụ thể:

1. Nhập vào biểu thức trung tố, ví dụ:

$$9 + 2 + 8 * 6$$

2. In ra biểu thức ở dạng hậu tố, ví dụ:  $9\ 2\ +\ 8\ 6\ *\ +$

3. Tính ra giá trị của biểu thức vừa nhập

Các hằng số là số nguyên, trong phạm vi từ 0  $\rightarrow$  99.

Toán tử bao gồm các phép toán cộng, trừ, nhân, chia lấy thương(/), chia lấy dư(%), đóng mở ngoặc.

### A. Phân tích cách làm và thuật toán.

Để chuyển từ biểu thức trung tố sang biểu thức hậu tố, ta đọc lần lượt từng ký tự trong biểu thức trung tố, giả sử ký tự đọc được là c:

- + Nếu c là số  $\rightarrow$  thêm vào biểu thức hậu tố
- + Nếu c là toán tử  $\rightarrow$  thực hiện thêm toán tử vào stack. Nếu stack rỗng, thêm c vào stack, nếu stack không rỗng, so sánh thứ tự ưu tiên của c với toán tử ở đỉnh stack, nếu độ ưu tiên của c  $\leq$  độ ưu tiên của toán tử ở đỉnh stack, lấy toán tử ở đỉnh ra và thêm vào biểu thức hậu tố, lặp lại việc so sánh trên cho đến khi độ ưu tiên của c  $\geq$  độ ưu tiên của toán tử ở đỉnh stack, sau đó thêm c vào stack.
- + Nếu c là toán tử (  $\rightarrow$  thêm luôn c vào stack.
- + Nếu c là toán tử )  $\rightarrow$  lấy lần lượt các toán tử trong stack ra và thêm vào biểu thức hậu tố cho đến khi gặp toán tử ) . Lưu ý, không thêm toán tử ) vào trong biểu thức hậu tố.
- + Khi duyệt hết biểu thức trung tố, nếu còn toán tử trong stack, thực hiện lấy lần lượt các toán tử ra và thêm vào biểu thức trung tố cho đến khi stack rỗng.

Vì các toán hạng có giá trị từ 0  $\rightarrow$  99 nên khi thêm vào biểu thức hậu tố, cần thêm các dấu cách để ngăn cách giữa các phần tử.

Để tính giá trị biểu thức hậu tố:

- + Thực hiện duyệt biểu thức hậu tố, nếu gặp toán hạng thì thêm toán hạng vào stack, nếu gặp toán tử, lấy hai toán hạng trong stack ra và tính toán phép tính theo toán tử tương ứng, kết quả của phép tính được lưu trở lại vào trong stack. Lặp lại cho đến khi duyệt hết biểu thức hậu tố, kết quả cuối cùng được lưu trong stack.

Thứ tự ưu tiên của các toán tử:

| Toán tử | Độ ưu tiên |
|---------|------------|
| (       | 0          |
| +       | 1          |
| -       | 1          |
| *       | 2          |
| /       | 2          |
| %       | 2          |
| )       | 3          |

## B. Mã nguồn

```
1 .data
2 btrungto: .space 256
3 bthauto: .space 256
4 nganxep: .space 256
5 arr: .space 256
6
7 string: .asciiz "\n"
8 message1: .asciiz "Bieu thuc trung to: "
9 message2: .asciiz "Bieu thuc hau to: "
10 message3: .asciiz "Ket qua bieu thuc vua nhap: "
11 message4: .asciiz "Nhap vao bieu thuc trung to: "
12 message5: .asciiz "MENU\n1.Chay chương trình\n2.Thoat chương trình\nBan chon?\n "
13 message6: .asciiz "Bieu thuc khong hop le.\n"
14 .text
15 main:la $a0, message5
16     li $v0, 4
17     syscall
18
19     li $v0, 5
20     syscall
21
22     beq $v0, 2, end_main
23
24     jal INPUT
25     nop
26     jal CHECK
27     nop
28     beq $v0, 0, next1
29     nop
30
31     jal OUTPUT1
32     nop
33     jal CONVERT
34     nop
35     jal OUTPUT2
36     jal CALCULATE
37     jal OUTPUT3
38     j main
39     nop
40 next1:la $a0, message6
```

nop Null operation : machine code is all zeroes

```

41     li $v0, 4
42     syscall
43
44     j main
45 end_main:
46     li $v0, 10
47     syscall
48
49 #Nhap bieu thuc trung to
50 INPUT:la $a0, message4
51     li $v0, 4
52     syscall
53
54     li $v0, 8
55     la $a0, bttrungto
56     la $a1, 256
57     syscall
58
59     jr $ra
60     nop
61 #In bieu thuc trung to ra man hinh
62 OUTPUT1:la $a0, message1
63     li $v0, 4
64     syscall
65
66     la $a0, bttrungto
67     li $v0, 4
68     syscall
69
70     jr $ra
71     nop
72 #In bieu thuc hau to ra man hinh
73 OUTPUT2: la $a0, message2
74     li $v0, 4
75     syscall
76
77     la $a0, bthauto
78     li $v0, 4
79     syscall
80

```

```

81     la $a0, string
82     li $v0, 4
83     syscall
84
85     jr $ra
86     nop
87 #In ket qua ra man hinh
88 OUTPUT3:la $a0, message3
89     li $v0, 4
90     syscall
91
92     lb $a0, -1($s3)
93     li $v0, 1
94     syscall
95
96     la $a0, string
97     li $v0, 4
98     syscall
99
100    jr $ra
101    nop
102 #Kiem tra tinh hop le cua bieu thuc vua nhap
103 CHECK:addi $sp, $sp, -4
104     sw $ra, 0($sp)
105     la $a0, bttrungto
106     li $t1, 0
107 load:add $t2, $t1, $a0
108     lb $t3, 0($t2)
109     beq $t3, 10, end_check
110     jal check_number
111     beq $v0, 1, continue
112     jal check_operator
113     beq $v0, 1, continue
114     jal check_space
115     beq $v0, 1, continue
116     j end_check
117
118 continue:
119     addi $t1, $t1, 1
120     j load

```

```

121
122 end_check:
123     lw $ra, 0($sp)
124     addi $sp, $sp, 4
125     jr $ra
126     nop
127 #CONVERT : chuyen bieu thuc trung to thanh bieu thuc hau to
128 CONVERT:addi $sp, $sp, -4
129     sw $ra, 0($sp)
130     la $s1, bttrungto
131     la $s2, bthauto
132     la $s3, nganxep
133     li $s6, 0
134     li $s7, 0
135     li $t7, -1
136 while1:add $s4, $s1, $s6
137     lb $t3, 0($s4)
138     beq $t3, 10, end_while1
139     jal check_number
140     beq $v0, 1, push_number
141 next3:beq $t3, 10, end_while1
142     jal check_operator
143     beq $v0, 1, check_before_push_operator
144 next2:addi $s6, $s6, 1
145     j while1
146 push_number:add $s4, $s2, $s7
147     sb $t3, 0($s4)
148     addi $s7, $s7, 1
149     addi $s6, $s6, 1
150     add $s4, $s1, $s6
151     lb $t3, 0($s4)
152     jal check_number
153     beq $v0, 1, push_number
154     li $t2, ' '
155     add $s4, $s2, $s7
156     sb $t2, 0($s4)
157     addi $s7, $s7, 1
158     j next3
159 check_before_push_operator:beq $s0, 0, push_operator1
160     beq $t7, -1, push_operator1

161     addi $t1, $t3, 0
162     addi $a0, $s0, 0
163 after_pop:beq $t7, -1, push_operator2
164     add $s4, $t7, $s3
165     lb $t3, 0($s4)
166     jal check_operator
167     ble $a0, $s0, pop_operator
168     j push_operator2
169 pop_operator:beq $t3, '(', before_pop_operator
170     add $s4, $s2, $s7
171     sb $t3, 0($s4)
172     addi $s7, $s7, 1
173     li $t2, ' '
174     add $s4, $s2, $s7
175     sb $t2, 0($s4)
176     addi $s7, $s7, 1
177     addi $t7, $t7, -1
178     j after_pop
179 before_pop_operator:addi $t7, $t7, -1
180     j push_operator2
181 push_operator1:addi $t7, $t7, 1
182     add $s4, $s3, $t7
183     sb $t3, 0($s4)
184     j next2
185 push_operator2:beq $t1, ')', after_pop2
186     addi $t7, $t7, 1
187     add $s4, $s3, $t7
188     sb $t1, 0($s4)
189     j next2
190 after_pop2:
191     add $s4, $t7, $s3
192     lb $t3, 0($s4)
193     addi $t7, $t7, -1
194     beq $t3, '(', push_operator3
195     add $s4, $s2, $s7
196     sb $t3, 0($s4)
197     addi $s7, $s7, 1
198     li $t2, ' '
199     add $s4, $s2, $s7
200     sb $t2, 0($s4)

```

```

201     addi $s7, $s7, 1
202     j after_pop2
203 push_operator3:
204     j next2
205 end_while1: beq $t7, -1, rt3
206     add $s4, $s3, $t7
207     lb $t3, 0($s4)
208     addi $t7, $t7, -1
209     add $s4, $s2, $s7
210     sb $t3, 0($s4)
211     addi $s7, $s7, 1
212     li $t2, ' '
213     add $s4, $s2, $s7
214     sb $t2, 0($s4)
215     addi $s7, $s7, 1
216     j end_while1
217 rt3: li $t2, '\0'
218     add $s4, $s2, $s7
219     sb $t2, 0($s4)
220     lw $ra, 0($sp)
221     addi $sp, $sp, 4
222     jr $ra
223     nop
224
225 #CACULATE
226 # $s2 : bieu thuc hau to
227 # $s3 : ngan xep
228 CALCULATE: addi $sp, $sp, -4
229     sw $ra, 0($sp)
230     li $s6, 0
231     li $s7, -4
232     la $t2, arr
233 while2: add $s4, $s2, $s6
234     lb $t3, 0($s4)
235     beq $t3, 0, end_while2
236     jal check_number
237     li $t9, -1
238     li $s5, 0
239     beq $v0, 1, before_convert_num
240 next4: beq $t3, 0, end_while2
241
242 jal check_operator
243 beq $v0, 1, ccl
244 next5: addi $s6, $s6, 1
245     j while2
246 before_convert_num: add $s4, $t2, $s5
247     sb $t3, 0($s4)
248     addi $s5, $s5, 1
249     addi $t9, $t9, 1
250     addi $s6, $s6, 1
251     add $s4, $s2, $s6
252     lb $t3, 0($s4)
253     jal check_number
254     beq $v0, 1, before_convert_num
255     jal convert_num
256     j next4
257 ccl: addi $s3, $s3, -2
258     lb $a0, ($s3)
259     lb $a1, 1($s3)
260     jal CAL
261     j next5
262 end_while2:
263     lw $ra, ($sp)
264     addi $sp, $sp, 4
265     jr $ra
266 #Check xem co phai ki tu space khong?
267 # $v0 = 0 -> khong la space
268 # $v0 = 1 -> la space
269 check_space:
270     bne $t3, ' ', check_space_false
271 check_space_true:
272     li $v0, 1
273     jr $ra
274     nop
275 check_space_false:
276     li $v0, 0
277     jr $ra
278     nop
279 #Check xem co phai toan tu khong?
280 # $v0 = 0 -> khong phai la toan tu
281 # $v0 = 1 -> la toan tu, $s0 luu tru thu tu uu tien cua toan tu

```



```

281 check_operator:
282     addi $sp, $sp, -28
283     sw   $s2, 24($sp)
284     sw   $s3, 20($sp)
285     sw   $s4, 16($sp)
286     sw   $s5, 12($sp)
287     sw   $s6, 8($sp)
288     sw   $s7, 4($sp)
289     sw   $t8, ($sp)
290     li   $s2, '+'
291     li   $s3, '-'
292     li   $s4, '*'
293     li   $s5, '/'
294     li   $s6, '%'
295     li   $s7, '('
296     li   $t8, ')'
297     beq  $t3, $s2, operator1
298     beq  $t3, $s3, operator1
299     beq  $t3, $s4, operator2
300     beq  $t3, $s5, operator2
301     beq  $t3, $s6, operator2
302     beq  $t3, $s7, operator0
303     beq  $t3, $t8, operator3
304     j    check_operator_false
305 operator0: li $s0, 0
306     j    check_operator_true
307 operator1: li $s0, 1
308     j    check_operator_true
309 operator2: li $s0, 2
310     j    check_operator_true
311 operator3: li $s0, 3
312 check_operator_true:
313     li   $v0, 1
314     j    rt1
315     nop
316 check_operator_false:
317     li   $v0, 0
318 rt1: lw   $s2, 24($sp)
319     lw   $s3, 20($sp)
320     lw   $s4, 16($sp)
321
322     lw   $s5, 12($sp)
323     lw   $s6, 8($sp)
324     lw   $s7, 4($sp)
325     lw   $t8, ($sp)
326     addi $sp, $sp, 28
327     jr   $ra
328     nop
329 #Check xem co phai so khong?
330 # $v0 = 0 -> khong la so
331 # $v0 = 1 -> la so
332 check_number:
333     addi $sp, $sp, -8
334     sw   $t8, 4($sp)
335     sw   $t9, ($sp)
336     li   $t8, '0'
337     li   $t9, '9'
338
339     beq  $t8, $t3, check_number_true
340     beq  $t9, $t3, check_number_true
341     bgt  $t8, $t3, check_number_false
342     bgt  $t9, $t3, check_number_false
343 check_number_true:
344     li   $v0, 1
345     j    rt2
346 check_number_false:
347     li   $v0, 0
348 rt2: lw   $t9, ($sp)
349     lw   $t8, 4($sp)
350     addi $sp, $sp, 8
351     jr   $ra
352 #Tinh gia tri bieu thuc
353 # $a0 : phan tu truoc
354 # $a1 : phan tu sau
355 # $t3 : op
356 # $a0 op $a1
357 CAL:
358 add_op: bne $t3, '+', minus_op
359     add  $v0, $a0, $a1
360     j    rt_cal

```

```

361 minus_op: bne $t3, '-', mul_op
362           sub $v0, $a0, $a1
363           j    rt_cal
364 mul_op:   bne $t3, '*', div_op
365           mult $a0, $a1
366           mflo $v0
367           j    rt_cal
368 div_op:   bne $t3, '/', divr_op
369           div $a0, $a1
370           mflo $v0
371           j    rt_cal
372 divr_op:  bne $t3, '%', rt_cal
373           div $a0, $a1
374           mfhi $v0
375 rt_cal:
376           sb   $v0, ($s3)
377           addi $s3, $s3, 1
378           jr   $ra
379 #Chuyen doi ky tu thanh so
380 # $t2 : luu ki tu
381 # $s3 : lu so tuong ung
382 # $t9 : bien dem cua mang luu ki tu
383 convert_num:
384           addi $sp, $sp, -8
385           sw   $t1, ($sp)
386           sw   $t5, 4($sp)
387           bne $t9, $0, twoNum
388           lb   $t1, 0($t2)
389           addi $t1, $t1, -48
390           j    push
391 twoNum:
392           lb   $t1, 0($t2)
393           addi $t1, $t1, -48
394           addi $t5, $0, 10
395           mult $t1, $t5
396           mflo $t1
397           lb   $t5, 1($t2)
398           addi $t5, $t5, -48
399           add  $t1, $t1, $t5
400 push: sb $t1, 0($s3)

401           lw   $t1, ($sp)
402           addi $sp, $sp, 8
403           j    rt4
404 end_loop2:
405 end_loop1:
406 rt4: jr $ra
407      nop
408
409

```

## C. Kết quả chạy mô phỏng

|                                      |   |
|--------------------------------------|---|
|                                      | <pre>MENU 1.Chay chuong trinh 2.Thoat chuong trinh Ban chon? 1 Nhap vao bieu thuc trung to: 2%3/4+5 Bieu thuc trung to: 2%3/4+5 Bieu thuc hau to: 2 3 % 4 / 5 + Ket qua bieu thuc vua nhap: 5</pre> |
| <input type="button" value="Clear"/> | <pre>MENU 1.Chay chuong trinh 2.Thoat chuong trinh Ban chon? 1 Nhap vao bieu thuc trung to: d+&amp;.r4 Bieu thuc khong hop le. MENU 1.Chay chuong trinh 2.Thoat chuong trinh Ban chon? 2</pre>      |
|                                      | <pre>MENU 1.Chay chuong trinh 2.Thoat chuong trinh Ban chon? 2  -- program is finished running --</pre>   |