

歷年作品

OurScheme(Scheme-Interpreter)

```

Welcome to OurScheme!(CYCUICE 10727219)
> (define a 3) (define b 4) (define c 5)
a defined
> b defined
> c defined
> ( * 10 ( + a b c ) )
120
> ( if ( > a b )
      ( + b c )
      ( * b c ) )
20
> (define name "Chih Chung Hsu" )
name defined
> ( string-append "CYCU" "ICE " name )
"CYCUICE Chih Chung Hsu"
> ( define ( Func x ) ( / 10.0 x ) )
Func defined
> ( Func 5 ) ( Func 0 )
2.000
> ERROR (division by zero) : /
> (define F1 ( lambda (x) (+ x x) ) )
F1 defined
> (define ( F2 x ) ( ( lambda (x) (+ x x) ) x ) )
F2 defined
> ( F1 a )
6
> ( F2 a )
6
> (define ( F3 y ) ( cond ( ( < y 10 ) "first" ) (else "second" ) ) )
F3 defined
> ( F3 9 )
"first"
> ( F3 11 )
"second"

( define F4 ( lambda ( a b c ) ( list a b c ) ) )
F4 defined
> ( list a b c )
( 3 4 5 )
> (F4 c b a )
( 5 4 3 )
>
(define d ( define d 5 ))
ERROR (level of DEFINE)
> ( + 5 "symbol" )
ERROR (+ with incorrect argument type) : "symbol"
> ( if ( = 7 4 ) "error" )
ERROR (no return value) : ( if
  ( =
    7
    4
  )
  "error"
)
> ( if ( if ( > 10 20 ) #t ) "ture" )
ERROR (unbound parameter) : ( if
  ( >
    10
    20
  )
  #t
)

```

以 C/C++ 語言實作之 scheme 語言直譯器，除了能夠運行基本的 scheme 語言指令之外，也能處理複雜 function 呼叫、lambda 的宣告，並能針對錯誤語法之指令加以處理並輸出錯誤訊息。

VI-Editor



The screenshot shows the VI-Editor interface. On the left, there is a window titled '輸出 - Vleditor (run)' which displays the output of the program. The output text is as follows:

```

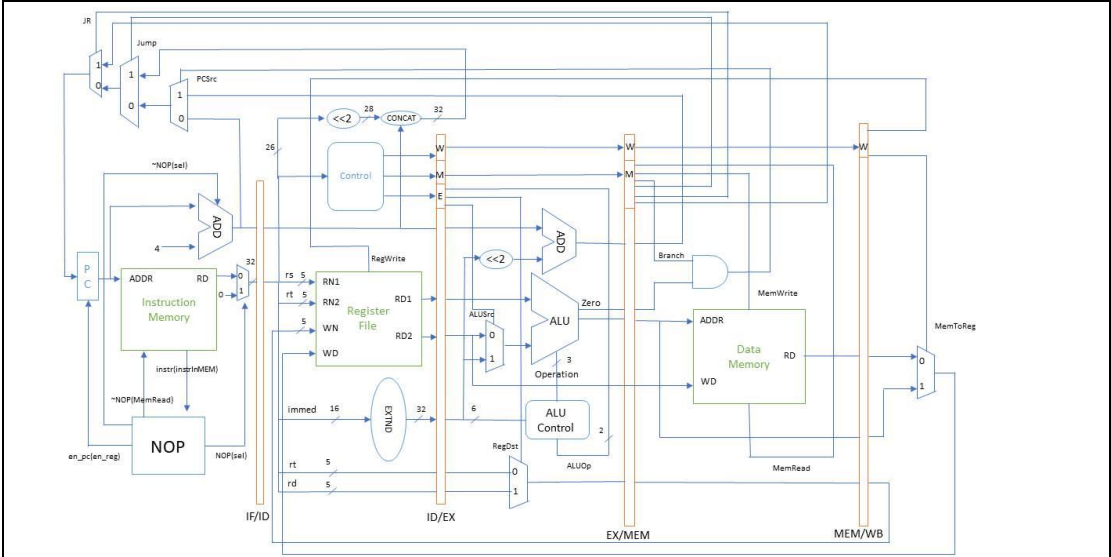
run:
歡迎使用中原資工Line editor (許志仲)...
> b
Buffer-0 0
> a
123456
.
> a
654654
.
> i
CYCUICE
3
.

```

On the right, the main editor window shows the source code of the program, which is a Scheme interpreter implementation. The code includes definitions for variables, functions, and macros, as well as error handling and user interface elements like 'Buffer-0' and 'page 2'.

以 java 語言實作的 VI 純文字編輯器，能夠簡易執行新增(a)、插入(i)、新頁(b)、印出(p)等簡易指令。

5 STAGE PIPELINE CPU



```

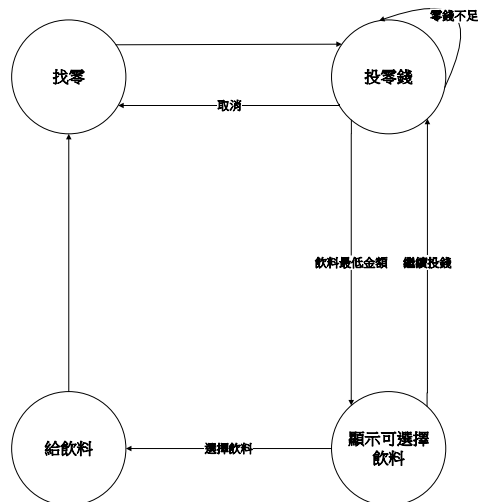
# 1, reading data: Mem[          0] => 2385444864
# 1, PC:          0
# 1, wd:          0
# 1, NOP
#
# 2, reg_file[15] =>          21 (Port 2)
# 2, reg_file[17] =>          2 (Port 1)
# 2, reading data: Mem[          4] => 305201155
# 2, PC:          4
# 2, LW
#
# 3, reg_file[17] =>          2 (Port 2)
# 3, PC:          8
# 3, BEQ
#
# 4, reg_file[ 0] =>          0 (Port 2)
# 4, reg_file[ 0] =>          0 (Port 1)
# 4, reading data: Mem[          2] =>      256
# 4, PC:          8
# 4, wd:          x
# 4, NOP
#
# 6, reg_file[15] <=      256 (Write)
# 5, PC:          8
# 5, wd:          256
# 5, NOP
#
# 6, reading data: Mem[          20] => 134217735
# 6, PC:          20
# 6, wd:          0
# 6, NOP
#
# 7, PC:          20
# 7, J
#
# 8, PC:          20
# 8, wd:          x
# 8, NOP
#
# 9, PC:          20
# 9, wd:          x
# 9, NOP
#

```

[illegible]

以 Verilog 語言實作模擬的 5 stage pipeline CPU，包含 32bit ALU、除法器、barrel shifter 等元件，可以執行 MIPS 指令。

自動販賣機



Result

CPU Time: sec(s), Memory: kilobyte(s)

```

exchange 0 dollars
coin 5, total 5 dollars

coin 10, total 15 dollars
tea(10) | coke(15)

exchange 15 dollars
coin 10, total 10 dollars
tea(10)

coin 10, total 20 dollars
tea(10) | coke(15) | coffee(20)

coke out
exchange 5 dollars
coin 10, total 10 dollars
tea(10)

coin 10, total 20 dollars
tea(10) | coke(15) | coffee(20)

not enough money
coin 10, total 30 dollars
tea(10) | coke(15) | coffee(20) | milk(25)

tea out
exchange 20 dollars
coin 50, total 50 dollars
tea(10) | coke(15) | coffee(20) | milk(25)

milk out
exchange 25 dollars
** VVP Stop(0) **
** Flushing output streams.
** Current simulation time is 230 ticks.
>
  
```

以 Verilog 語言實作，使用有限狀態機實踐的自動販賣機，共有投零錢、顯示、給飲料、找零等四階段功能。

```

SICXE_input.txt - 記事本
檔案(F) 編輯(E) 格式(O) 檢視(V) 說明
COPY      START 0
          var1   RESW    1
          var2   RESB   12
          var3   EQU    12
add var1
+addf var2
addr a,b
+and var1
clear b
comp 0
+compf var3
compr b,a
.this is comment
div var1
divf var1
divr l,s
fix
float
END      FIRST

```

```

C:\Users\JohnnyHsu\Documents\GitHub\CYCUICE_SP\Cross Assembler\10727219.exe
SP Project2
Please choose
1. SIC, 2. SIC/XE, 0 quit
Input :2
1
Please enter FileName : SICXE_input.txt
f1

```

Line	Location	Source code	Object code
5	0000	COPY START 0	
10	0000	var1 RESW 1	
15	0003	var2 RESB 12	
20	000C	var3 EQU 12	
25	000F	add var1	1B2FEE
30	0012	+ addf var2	5B100003
35	0016	addr a, b	9003
40	0018	+ and var1	43100000
45	001C	clear b	B430
50	001E	comp 0	2B0000
55	0021	+ compf var3	8B10000C
60	0025	compr b, a	A030
65		.this is comment	
70	0027	div var1	272FD6
75	002A	divf var1	672FD3
80	002D	divr l, s	9C24
85	002F	fix	C4
90	0030	float	C0
95		END FIRST	

```

Please choose

```

以 C/C++ 語言實作之 SIC/XE Assembler，以 2 Pass 的作法將 input 指令進行切 token、翻譯成 object code 的步驟。

```

input1.txt - 記事本
檔案(F) 編輯(E) 格式(O) 檢視(V) 說明
3
123412512345

out_input1.txt - 記事本
檔案(F) 編輯(E) 格式(O) 檢視(V) 說明
-----FIFO-----
1      1      F
2      21     F
3      321    F
4      432    F
1      143    F
2      214    F
5      521    F
1      521
2      521
3      352    F
4      435    F
5      435
Page Fault = 9  Page Replaces = 6  Page Frames = 3

-----LRU-----
1      1      F
2      21     F
3      321    F
4      432    F
1      143    F
2      214    F
5      521    F
1      152
2      215
3      321    F
4      432    F
5      543    F
Page Fault = 10 Page Replaces = 7  Page Frames = 3

```

以 C/C++ 語言實作 memory 中的六種分頁法包含 FIFO、LRU、LFU+FIFO、MFU+FIFO、LFU+LRU、MFU+LRU 等方法。

Show Hand

```

S4 S6 S8 S7 S5
-1
HQ DQ C2 S2 CQ
-1
Straight flush (Eight of Spades) wins over full house (Queen of Hearts).
-----
Process exited after 115 seconds with return value 0
請按任意鍵繼續 . . . █

```

以 C/C++ 語言實作的簡易撲克牌遊戲，根據兩位玩家輸入的牌組進行比對，根據擁有的組合和點數大小分出勝負。

CPU Scheduling

input1.txt - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明

ID	CPU	Burst	Arrival Time	Priority
6	1			
5	6	26	13	
13	1	7	2	
6	5	1	7	
27	6	3	7	
2	3	30	13	
1	2	13	5	
9	4	1	6	
10	8	2	13	
0	4	36	1	
8	2	23	12	
7	1	3	16	
29	6	20	8	
4	3	18	10	
20	3	15	14	
3	4	22	3	

out_input1.txt - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明

```
-997996D6666R11KKK444RR88RRR333322T0000TTTTT555555AAAAAAA
== PPRR==
-99996RD6R611R6RR4TT3333TTTT4488A500002A52A52A5A5AAKKK7
== HRRN==
-6666679999DRRRRRR11AAAAAAAKKK444883333TTTTTT222555550000
=====
```

waiting ID	FCFS	RR	SRTF	PPRR	HRRN
0	19	18	0	0	19
1	13	8	0	0	5
2	22	19	2	14	16
3	18	25	6	0	14
4	13	19	0	11	13
5	20	27	19	21	23
6	0	15	6	11	0
7	15	2	0	55	3
8	21	14	0	9	11
9	5	13	1	0	6
10	8	37	49	45	18
13	18	3	0	0	4
20	13	17	0	40	13
27	16	28	19	10	9
29	14	31	19	4	20

=====

以 C/C++ 語言實作模擬 CPU 排成演算法，包括 FCFS、Round Robin、SRTF、PPRR、HRRN 等五種 method。