程式人 月刊 雜誌

Programmer

讀書做善事、寫書做公益 – 歡迎程式人認養專欄或捐出您的網誌

參考價： NT 50 元，如果您喜歡本雜誌，請將書款捐贈公益團體

羅慧夫顱顏基金會 彰化銀行 (009) 帳號： 5234-01-41778-800

0123778

愛心條碼

程式人雜誌

2015 年 7 月

本期焦點：向 Nand2Tetris 學習電腦硬體設計

程式人雜誌

- 公益資訊

前言

編輯小語

最近小編發現了一個很棒的網路公開課，是 coursera 課程網站上的 From Nand to Tetris / Part I 這門課，於是我去修了這門課並作了習題，這是我真正修習的第一門網路公開課。

這門課程是教授學生如何從一個最基礎的 nand 邏輯閘開始，一路往上建構出所有基礎元件，像是 and, or, xor, not, MUX, DMUX, Adder, Memory 等等，接著建構出 CPU 與整台電腦，然後再學習如何在建構出這台電腦上的『組譯器、編譯器、作業系統』等等，最後在這個具備軟硬體的電腦上寫一個小型的方塊遊戲。

我覺得這門課非常棒，因此小編將整個修課的過程與心得分享給大家，也利用這個機會將這門『神級課程』介紹給大家認識。

----（「少年科技人雜誌」與「程式人雜誌」編輯 - 陳鍾誠）

授權聲明

請注意這點，避免侵權。 在此我們僅按著作權法中的合理使用原則，擷取幾張圖片，若您是該作品擁有者且認為此擷取行為不妥，請告訴我們，我們會配合移除。

本期焦點：向 **Nand2Tetris** 學習電腦硬體設計

Nand2Tetris -- 教您設計整台電腦的一門課

在上個月的『少年科技人雜誌』當中，我紀錄了自己修習 nand2tetris 這門 Coursera 網站上 MOOC 課程的過程，並且試圖將一台電腦從下到上設計所需要的基本知識講述清楚。

但是、由於 nand2tetris 網站上的 下列聲明 ，我決定不把自己的作業程式碼刊登出來。

> Code Posting Policy
> We developed this course and made all its materials freely availa ble because we want to help people learn applied computer science on their own terms. We believe that students and self-learners w ho set out to do the hardware and software projects should have t he benefit and challenge of doing original work, without seeing p ublished solutions.

Therefore, we request that you don't post solutions publicly on the web, e.g. in blogs or forums. If your course instructor or organizer creates a private space in which work can be shared outside the public domain, that's fine. Likewise, you can share your work with others using a password-protected space, if it's permitted by the specific course in which you are enrolled.

Please use your judgment and help ensure that many more students, like you, will not be denied the thrill of original work and self-discovery.

Thx Noam Nisan and Shimon Schocken

雖然如此，不過網路上已經有不少 nand2tetris 作業的程式碼，您只要 在 google 中打入 nand2tetris github 就可以找到這些作業了。

但是，對於程式人而言，有文章卻沒有程式可以對照閱讀，畢竟是個遺憾！

有鑑於此，我決定將自己在 nand2tetris 的作業改寫，從 nand2tetris 自製的硬體描述語言 HackHDL 改寫為 Verilog ，這樣不僅沒有公佈作業答案，也順便讓大家能夠瞭解如何用真正的硬體描述語言來設計處理器與整台電腦的方法，因此我們決定在本期當中用 Verilog 版的 nand2tetris 來說明電腦硬體的設計原理。

希望這樣的作法會對想要瞭解電腦硬體設計的程式人會有所幫助！

現在，就讓我們展開這趟旅程吧！

Nand2Tetris 硬體部份 -- 從邏輯閘到處理器

在 nand2tetris 的硬體部份，也就是課程 Part I 的作業上面，企圖導引學員從一個 nand 閘開始，一路經過 and, or, not, mux, dmux, adder, ALU, register, memory, CPU 到整台電腦，以便讓學習者能完整的理解一台電腦，並且自己動手實作出來。

以下是nand2tetris 的硬體部份的作業內容安排。

The Hack chip-set and hardware platform

Elementary logic gates

- Nand
- Not
- And
- Or
- Xor
- Mux
- Dmux
- Not16
- And16
- Or16
- Mux16
- Or8Way
- Mux4Way16
- Mux8Way16
- DMux4Way
- DMux8Way

done

Combinational chips

- HalfAdder
- FullAdder
- Add16
- Inc16
- ALU

done

Sequential chips

- DFF
- Bit
- Register
- RAM8
- RAM64
- RAM512
- RAM4K
- RAM16K
- PC

done

Computer Architecture

- Memory
- CPU
- Computer

this lecture

圖、nand2tetris 硬體部份的作業內容

Nand2tetris 課程中採用的硬體描述語言是課程專用的 HackHDL，其軟體也是為該課程專門設計的 HardEmulator，但是在本期雜誌中，我們將改用 Verilog 來實作。

如果您想瞭解如何用 HackHDL 實作，您可以參考以下這位修過 nand2tetris 的 havivha 同學在 github 上公佈的作業解答。

- https://github.com/havivha/Nand2Tetris

如果您想要先瞭解 nand2tetrix 課程背後所需要的基本知識，請參考上個月的少年科技人雜誌。

- NandToTetrix 慕課記 -- 從邏輯閘到方塊遊戲

如果您想學習 verilog 語法，或者瞭解如何用 verilog 設計其他處理器，也可以參考筆者的下列書籍與網頁。

- 陳鍾誠 / 教科書 / 計算機結構
- 免費電子書：Verilog 電路設計

在本期雜誌中，我們將學習如何用 verilog 設計出 nand2tetris 中的處理器 HackCPU 與整台電腦

HackComputer 。

現在、請跟著我一起重新體會『如何用 Verilog 重新詮釋 nand2tetris 硬體部份的課程』吧！

Nand2Tetris 第一週 -- 自製邏輯元件

我們將第一週的習題主要部份放在 gate.v , gate16.v 與 mux.v 這些 verilog 程式檔中，然後再分別寫測試檔去測試這些程式。

我們使用的測試工具是 icarus verilog 。

基本邏輯閘

程式模組： gate.v

```
module Nand(input a, b, output out);
  nand g1(out, a, b);
endmodule
```

```
module Not(input in, output out);
  Nand g1(in, in, out);
endmodule

module Or(input a, b, output out);
  Not g1(a, nota);
  Not g2(b, notb);
  Nand g3(nota, notb, out);
endmodule

module Xor(input a, b, output out);
  Nand g1(a, b, AnandB);
  Or   g2(a, b, AorB);
  And  g3(AnandB, AorB, out);
endmodule
```

```
module And(input a, b, output out);
  Nand g1(a, b, AnandB);
  Nand g2(AnandB, AnandB, out);
endmodule

module Or8Way(input[7:0] in, output out);
  Or g1(in[7], in[6], or76);
  Or g2(in[5], in[4], or54);
  Or g3(in[3], in[2], or32);
  Or g4(in[1], in[0], or10);
  Or g5(or76, or54, or74);
  Or g6(or32, or10, or30);
  Or g7(or74, or30, out);
endmodule
```

測試程式： gate_test.v

```
`include "gate.v"

module main;
reg a, b;
wire abNand, aNot, abAnd, abOr, abXor;

Not  g1(a, aNot);
Nand g2(a, b, abNand);
And  g3(a, b, abAnd);
Or   g4(a, b, abOr);
Xor  g5(a, b, abXor);

initial
begin
  $monitor("%4dns a=%d b=%d aNot=%d abNand=%d abAnd=%d abOr=%d ab
Xor=%d", $stime, a, b, aNot, abNand, abAnd, abOr, abXor);
```

```
  a  = 0;
  b  = 0;
end

always #50 begin
  a = a+1;
end

always #100 begin
  b = b+1;
end

initial #500 $finish;

endmodule
```

測試結果

```
D:\Dropbox\cccweb\db\n2t>iverilog gate_test.v -o gate_test

D:\Dropbox\cccweb\db\n2t>vvp gate_test
   0ns a=0 b=0 aNot=1 abNand=1 abAnd=0 abOr=0 abXor=0
  50ns a=1 b=0 aNot=0 abNand=1 abAnd=0 abOr=1 abXor=1
 100ns a=0 b=1 aNot=1 abNand=1 abAnd=0 abOr=1 abXor=1
 150ns a=1 b=1 aNot=0 abNand=0 abAnd=1 abOr=1 abXor=0
 200ns a=0 b=0 aNot=1 abNand=1 abAnd=0 abOr=0 abXor=0
 250ns a=1 b=0 aNot=0 abNand=1 abAnd=0 abOr=1 abXor=1
 300ns a=0 b=1 aNot=1 abNand=1 abAnd=0 abOr=1 abXor=1
 350ns a=1 b=1 aNot=0 abNand=0 abAnd=1 abOr=1 abXor=0
 400ns a=0 b=0 aNot=1 abNand=1 abAnd=0 abOr=0 abXor=0
 450ns a=1 b=0 aNot=0 abNand=1 abAnd=0 abOr=1 abXor=1
 500ns a=0 b=1 aNot=1 abNand=1 abAnd=0 abOr=1 abXor=1
```

16 位元邏輯閘

程式模組： gate16.v

```
`include "gate.v"

module Not16(input[15:0] in, output[15:0] out);
  Not g15(in[15], out[15]);
  Not g14(in[14], out[14]);
  Not g13(in[13], out[13]);
  Not g12(in[12], out[12]);
  Not g11(in[11], out[11]);
  Not g10(in[10], out[10]);
  Not g09(in[9], out[9]);
  Not g08(in[8], out[8]);
  Not g07(in[7], out[7]);
  Not g06(in[6], out[6]);
```

```
  Not g05(in[5], out[5]);
  Not g04(in[4], out[4]);
  Not g03(in[3], out[3]);
  Not g02(in[2], out[2]);
  Not g01(in[1], out[1]);
  Not g00(in[0], out[0]);
endmodule

module And16(input[15:0] a, b, output[15:0] out);
  And g15(a[15], b[15], out[15]);
  And g14(a[14], b[14], out[14]);
  And g13(a[13], b[13], out[13]);
  And g12(a[12], b[12], out[12]);
  And g11(a[11], b[11], out[11]);
  And g10(a[10], b[10], out[10]);
  And g09(a[9], b[9], out[9]);
```

```
  And g08(a[8], b[8], out[8]);
  And g07(a[7], b[7], out[7]);
  And g06(a[6], b[6], out[6]);
  And g05(a[5], b[5], out[5]);
  And g04(a[4], b[4], out[4]);
  And g03(a[3], b[3], out[3]);
  And g02(a[2], b[2], out[2]);
  And g01(a[1], b[1], out[1]);
  And g00(a[0], b[0], out[0]);
endmodule

module Or16(input[15:0] a, b, output[15:0] out);
  Or g15(a[15], b[15], out[15]);
  Or g14(a[14], b[14], out[14]);
  Or g13(a[13], b[13], out[13]);
  Or g12(a[12], b[12], out[12]);
```

```
  Or g11(a[11], b[11], out[11]);
  Or g10(a[10], b[10], out[10]);
  Or g09(a[9], b[9], out[9]);
  Or g08(a[8], b[8], out[8]);
  Or g07(a[7], b[7], out[7]);
  Or g06(a[6], b[6], out[6]);
  Or g05(a[5], b[5], out[5]);
  Or g04(a[4], b[4], out[4]);
  Or g03(a[3], b[3], out[3]);
  Or g02(a[2], b[2], out[2]);
  Or g01(a[1], b[1], out[1]);
  Or g00(a[0], b[0], out[0]);
endmodule
```

測試程式： gate16_test.v

```
`include "gate16.v"

module main;
reg  [15:0] a,b;
wire [15:0] aNot, abAnd, abOr;

Not16  g1(a, aNot);
And16  g2(a, b, abAnd);
Or16   g3(a, b, abOr);

initial
begin
  $monitor("a  =%b\nb  =%b\nnot=%b\nand=%b\nor =%b", a, b, aNot,
abAnd, abOr);
  a  = 16'b0011;
```

```
  b  = 16'b0101;
    $finish;
end

endmodule
```

測試結果

```
D:\Dropbox\cccweb\db\n2t>iverilog gate16_test.v -o gate16_test

D:\Dropbox\cccweb\db\n2t>vvp gate16_test
a  =0000000000000011
b  =0000000000000101
not=1111111111111100
and=0000000000000001
or =0000000000000111
```

多工器與解多工器 **(MUX and DMUX)**

程式模組： mux.v

```
`include "gate16.v"

module Mux(input a, b, sel, output out);
  Not g1(sel, nsel);
  And g2(a, nsel, o1);
  And g3(b, sel, o2);
  Or  g4(o1, o2, out);
endmodule

module DMux(input in, sel, output a, b);
  Not g1(sel, nsel);
  And g2(nsel, in, a);
  And g3(sel,  in, b);
```

```
endmodule

module Mux16(input[15:0] a, b, input sel, output[15:0] out);
  Mux g15(a[15], b[15], sel, out[15]);
  Mux g14(a[14], b[14], sel, out[14]);
  Mux g13(a[13], b[13], sel, out[13]);
  Mux g12(a[12], b[12], sel, out[12]);
  Mux g11(a[11], b[11], sel, out[11]);
  Mux g10(a[10], b[10], sel, out[10]);
  Mux g09(a[9],  b[9],  sel, out[9]);
  Mux g08(a[8],  b[8],  sel, out[8]);
  Mux g07(a[7],  b[7],  sel, out[7]);
  Mux g06(a[6],  b[6],  sel, out[6]);
  Mux g05(a[5],  b[5],  sel, out[5]);
  Mux g04(a[4],  b[4],  sel, out[4]);
  Mux g03(a[3],  b[3],  sel, out[3]);
```

```
  Mux g02(a[2],  b[2],  sel, out[2]);
  Mux g01(a[1],  b[1],  sel, out[1]);
  Mux g00(a[0],  b[0],  sel, out[0]);
endmodule

module Mux4Way16(input[15:0] a,b,c,d, input[1:0] sel, output[15:0
] out);
  wire [15:0] outab, outcd;
  Mux16 g1(a, b, sel[0], outab);
  Mux16 g2(c, d, sel[0], outcd);
  Mux16 g3(outab, outcd, sel[1], out);
endmodule

module Mux8Way16(input[15:0] a,b,c,d,e,f,g,h, input[2:0] sel, out
put[15:0] out);
  wire [15:0] outad, outeh;
```

```
  Mux4Way16 g1(a, b, c, d, sel[1:0], outad);
  Mux4Way16 g2(e, f, g, h, sel[1:0], outeh);
  Mux16     g3(outad, outeh, sel[2], out);
endmodule

module DMux4Way(input in, input[1:0] sel, output a,b,c,d);
  Not  g1(sel[1], nsel1);
  Not  g2(sel[0], nsel0);
  And  g3(nsel1,  nsel0,  sel00);
  And  g4(nsel1,  sel[0], sel01);
  And  g5(sel[1], nsel0,  sel10);
  And  g6(sel[1], sel[0], sel11);
  DMux g7(in, sel00, d0, a);
  DMux g8(in, sel01, d1, b);
  DMux g9(in, sel11, d2, d);
  DMux g10(in, sel10, d3, c);
```

```
endmodule

module DMux8Way(input in, input[2:0] sel, output a,b,c,d,e,f,g,h)
;
  Not g1(sel[2], nsel2);
  And g2(in, sel[2], s2h);
  And g3(in, nsel2,  s2l);
  DMux4Way g4(s2h, sel[1:0], e, f, g, h);
  DMux4Way g5(s2l, sel[1:0], a, b, c, d);
endmodule
```

測試程式： mux_test.v

```
`include "mux.v"

module main;
```

```
reg[15:0] a, b, c, d, e, f, g, h;
reg[2:0]  sel;
wire[15:0] mux2, mux4, mux8;
wire mux01, dmux0, dmux1;

Mux       g1(1'b0, 1'b1, sel[2], mux01);
DMux      g2(a[0], sel[2], dmux0, dmux1);
Mux16     g4(a, b, sel[0], mux2);
Mux4Way16 g5(a, b, c, d, sel[1:0], mux4);
Mux8Way16 g6(a, b, c, d, e, f, g, h, sel[2:0], mux8);

initial
begin
  $monitor("%4dns sel=%d mux2=%x mux4=%x mux8=%x", $stime, sel, m
ux2, mux4, mux8);
  a  = 16'h0;
```

```
  b  = 16'h1;
  c  = 16'h2;
  d  = 16'h3;
  e  = 16'h4;
  f  = 16'h5;
  g  = 16'h6;
  h  = 16'h7;
    sel = 0;
end

always #50 begin
  sel=sel+1;
end

initial #500 $finish;
```

```
endmodule
```

測試結果

```
D:\Dropbox\cccweb\db\n2t>iverilog mux_test.v -o mux_test

D:\Dropbox\cccweb\db\n2t>vvp mux_test
   0ns sel=0 mux2=0000 mux4=0000 mux8=0000
  50ns sel=1 mux2=0001 mux4=0001 mux8=0001
 100ns sel=2 mux2=0000 mux4=0002 mux8=0002
 150ns sel=3 mux2=0001 mux4=0003 mux8=0003
 200ns sel=4 mux2=0000 mux4=0000 mux8=0004
 250ns sel=5 mux2=0001 mux4=0001 mux8=0005
 300ns sel=6 mux2=0000 mux4=0002 mux8=0006
 350ns sel=7 mux2=0001 mux4=0003 mux8=0007
 400ns sel=0 mux2=0000 mux4=0000 mux8=0000
```

```
450ns sel=1 mux2=0001 mux4=0001 mux8=0001
500ns sel=2 mux2=0000 mux4=0002 mux8=0002
```

結語

這些閘的設計並不太困難，原理部份請參考下列文章或數位邏輯教科書。

- 少年科技人雜誌 / 2015年6月號 / Nand2Tetris 第 1 週 -- 布林函數

Nand2Tetris 第二週 -- 自製算術元件

算術元件部份包含『半加器、全加器、16位元加法器與 ALU 等等』，我們的 verilog 版模組與測試程式如下所示。

程式模組： alu.v

```
`include "mux.v"

module HalfAdder(input a,b, output sum, carry);
```

```
  Xor g1(a, b, sum);
  And g2(a, b, carry);
endmodule

module FullAdder(input a,b,c, output sum, carry);
  wire s1, c1, c2;
  Xor g1(a,  b,  s1);
  Xor g2(s1, c,  sum);
  And g3(a,  b,  c1);
  And g4(s1, c,  c2);
  Xor g5(c2, c1, carry);
endmodule

module Add16(input[15:0] a,b, output[15:0] out);
  wire [15:0] c;
  FullAdder g01(a[0],  b[0],  1'b0,  out[0],  c[0]);
```

```
  FullAdder g02(a[1],  b[1],  c[0],  out[1],  c[1]);
  FullAdder g03(a[2],  b[2],  c[1],  out[2],  c[2]);
  FullAdder g04(a[3],  b[3],  c[2],  out[3],  c[3]);
  FullAdder g05(a[4],  b[4],  c[3],  out[4],  c[4]);
  FullAdder g06(a[5],  b[5],  c[4],  out[5],  c[5]);
  FullAdder g07(a[6],  b[6],  c[5],  out[6],  c[6]);
  FullAdder g08(a[7],  b[7],  c[6],  out[7],  c[7]);
  FullAdder g09(a[8],  b[8],  c[7],  out[8],  c[8]);
  FullAdder g10(a[9],  b[9],  c[8],  out[9],  c[9]);
  FullAdder g11(a[10], b[10], c[9],  out[10], c[10]);
  FullAdder g12(a[11], b[11], c[10], out[11], c[11]);
  FullAdder g13(a[12], b[12], c[11], out[12], c[12]);
  FullAdder g14(a[13], b[13], c[12], out[13], c[13]);
  FullAdder g15(a[14], b[14], c[13], out[14], c[14]);
  FullAdder g16(a[15], b[15], c[14], out[15], c[15]);
endmodule
```

```
module Inc16(input[15:0] in, output[15:0] out);
  Add16 g1(in, 16'h1, out);
endmodule

//  x[16], y[16],  // 16-bit inputs
//  zx, // zero the x input?
//  nx, // negate the x input?
//  zy, // zero the y input?
//  ny, // negate the y input?
//  f,  // compute out = x + y (if 1) or x & y (if 0)
//  no; // negate the out output?
//  out[16], zr, ng;  // zr:zero, ng:negative

module ALU(input[15:0] x, y, input zx,nx,zy,ny,f,no, output[15:0]
 out, output zr, ng);
```

```
  wire[15:0] x1, notx1, x2, y1, noty1, y2, andxy, addxy, o1, noto
1, o2;
    wire orLow, orHigh, notzr;

  Mux16 g1(x,  16'b0, zx, x1);   // if (zx == 1) set x = 0
  Not16 g2(x1, notx1);
  Mux16 g3(x1, notx1, nx, x2);   // if (nx == 1) set x = !x

  Mux16 g4(y,  16'b0, zy, y1);   // if (zy == 1) set y = 0
  Not16 g5(y1, noty1);
  Mux16 g6(y1, noty1, ny, y2);   // if (ny == 1) set y = !y

  Add16 g7(x2, y2, addxy);       // addxy = x + y
  And16 g8(x2, y2, andxy);       // andxy = x & y

  Mux16 g9(andxy, addxy, f, o1); // if (f == 1)  set out = x + y
```

```
else set out = x & y
  Not16 g10(o1, noto1);

  Mux16 g11(o1, noto1, no, o2);   // if (no == 1) set out = !out

  // o2 就是 out, 但必須中間節點才能再次當作輸入，所以先用 o2。
  And16 g12(o2, o2, out);
  Or8Way g13(out[7:0],  orLow);  // orLow = Or(out[0..7]);
  Or8Way g14(out[15:8], orHigh);// orHigh = Or(out[8..15]);
  Or    g15(orLow, orHigh, notzr);  // nzr = Or(out[0..15]);
  Not   g16(notzr, zr);             // zr = !nzr
  And   g17(o2[15], o2[15], ng);   // ng = out[15]
  And16 g18(o2, o2, out);
endmodule
```

測試程式： gate_test.v

```
`include "alu.v"

module main;
reg signed[15:0] x, y;
reg zx,nx,zy,ny,f,no;
wire signed[15:0] out;
wire zr, ng;

ALU alu1(x,y, zx,nx,zy,ny,f,no, out, zr,ng);

initial
begin
  $monitor("%4dns x=%d y=%d zx=%b nx=%b zy=%b ny=%b f=%b no=%b ou
t=%d zr=%b ng=%b", $stime, x, y, zx, nx, zy, ny, f, no, out, zr,
ng);
    x = 9;
```

```
    y = 15;
    zx = 0;
    nx = 0;
    zy = 0;
    ny = 0;
    f  = 0;
    no = 0;
end

always #320 begin
  zx = zx+1;
end

always #160 begin
  nx = nx+1;
end
```

```
always #80 begin
  zy = zy+1;
end

always #40 begin
  ny = ny+1;
end

always #20 begin
  f = f+1;
end

always #10 begin
  no = no+1;
end
```

```
initial #640 $finish;

endmodule
```

測試結果

```
D:\Dropbox\cccweb\db\n2t>iverilog alu_test.v -o alu_test

D:\Dropbox\cccweb\db\n2t>vvp alu_test
   0ns x=     9 y=    15 zx=0 nx=0 zy=0 ny=0 f=0 no=0 out=     9
zr=0 ng=0
  10ns x=     9 y=    15 zx=0 nx=0 zy=0 ny=0 f=0 no=1 out=   -10
zr=0 ng=1
  20ns x=     9 y=    15 zx=0 nx=0 zy=0 ny=0 f=1 no=0 out=    24
zr=0 ng=0
```

```
  30ns x=     9 y=    15 zx=0 nx=0 zy=0 ny=0 f=1 no=1 out=   -25
zr=0 ng=1
  40ns x=     9 y=    15 zx=0 nx=0 zy=0 ny=1 f=0 no=0 out=     0
zr=1 ng=0
  50ns x=     9 y=    15 zx=0 nx=0 zy=0 ny=1 f=0 no=1 out=    -1
zr=0 ng=1
  60ns x=     9 y=    15 zx=0 nx=0 zy=0 ny=1 f=1 no=0 out=    -7
zr=0 ng=1
  70ns x=     9 y=    15 zx=0 nx=0 zy=0 ny=1 f=1 no=1 out=     6
zr=0 ng=0
  80ns x=     9 y=    15 zx=0 nx=0 zy=1 ny=0 f=0 no=0 out=     0
zr=1 ng=0
  90ns x=     9 y=    15 zx=0 nx=0 zy=1 ny=0 f=0 no=1 out=    -1
zr=0 ng=1
 100ns x=     9 y=    15 zx=0 nx=0 zy=1 ny=0 f=1 no=0 out=     9
zr=0 ng=0
```

```
 110ns x=     9 y=    15 zx=0 nx=0 zy=1 ny=0 f=1 no=1 out=   -10
zr=0 ng=1
 120ns x=     9 y=    15 zx=0 nx=0 zy=1 ny=1 f=0 no=0 out=     9
zr=0 ng=0
 130ns x=     9 y=    15 zx=0 nx=0 zy=1 ny=1 f=0 no=1 out=   -10
zr=0 ng=1
 140ns x=     9 y=    15 zx=0 nx=0 zy=1 ny=1 f=1 no=0 out=     8
zr=0 ng=0
 150ns x=     9 y=    15 zx=0 nx=0 zy=1 ny=1 f=1 no=1 out=    -9
zr=0 ng=1
 160ns x=     9 y=    15 zx=0 nx=1 zy=0 ny=0 f=0 no=0 out=     6
zr=0 ng=0
 170ns x=     9 y=    15 zx=0 nx=1 zy=0 ny=0 f=0 no=1 out=    -7
zr=0 ng=1
 180ns x=     9 y=    15 zx=0 nx=1 zy=0 ny=0 f=1 no=0 out=     5
zr=0 ng=0
```

```
 190ns x=     9 y=    15 zx=0 nx=1 zy=0 ny=0 f=1 no=1 out=    -6
zr=0 ng=1
 200ns x=     9 y=    15 zx=0 nx=1 zy=0 ny=1 f=0 no=0 out=   -16
zr=0 ng=1
 210ns x=     9 y=    15 zx=0 nx=1 zy=0 ny=1 f=0 no=1 out=    15
zr=0 ng=0
 220ns x=     9 y=    15 zx=0 nx=1 zy=0 ny=1 f=1 no=0 out=   -26
zr=0 ng=1
 230ns x=     9 y=    15 zx=0 nx=1 zy=0 ny=1 f=1 no=1 out=    25
zr=0 ng=0
 240ns x=     9 y=    15 zx=0 nx=1 zy=1 ny=0 f=0 no=0 out=     0
zr=1 ng=0
 250ns x=     9 y=    15 zx=0 nx=1 zy=1 ny=0 f=0 no=1 out=    -1
zr=0 ng=1
 260ns x=     9 y=    15 zx=0 nx=1 zy=1 ny=0 f=1 no=0 out=   -10
zr=0 ng=1
```

```
 270ns x=     9 y=    15 zx=0 nx=1 zy=1 ny=0 f=1 no=1 out=     9
zr=0 ng=0
 280ns x=     9 y=    15 zx=0 nx=1 zy=1 ny=1 f=0 no=0 out=   -10
zr=0 ng=1
 290ns x=     9 y=    15 zx=0 nx=1 zy=1 ny=1 f=0 no=1 out=     9
zr=0 ng=0
 300ns x=     9 y=    15 zx=0 nx=1 zy=1 ny=1 f=1 no=0 out=   -11
zr=0 ng=1
 310ns x=     9 y=    15 zx=0 nx=1 zy=1 ny=1 f=1 no=1 out=    10
zr=0 ng=0
 320ns x=     9 y=    15 zx=1 nx=0 zy=0 ny=0 f=0 no=0 out=     0
zr=1 ng=0
 330ns x=     9 y=    15 zx=1 nx=0 zy=0 ny=0 f=0 no=1 out=    -1
zr=0 ng=1
 340ns x=     9 y=    15 zx=1 nx=0 zy=0 ny=0 f=1 no=0 out=    15
zr=0 ng=0
```

```
 350ns x=     9 y=    15 zx=1 nx=0 zy=0 ny=0 f=1 no=1 out=   -16
zr=0 ng=1
 360ns x=     9 y=    15 zx=1 nx=0 zy=0 ny=1 f=0 no=0 out=     0
zr=1 ng=0
 370ns x=     9 y=    15 zx=1 nx=0 zy=0 ny=1 f=0 no=1 out=    -1
zr=0 ng=1
 380ns x=     9 y=    15 zx=1 nx=0 zy=0 ny=1 f=1 no=0 out=   -16
zr=0 ng=1
 390ns x=     9 y=    15 zx=1 nx=0 zy=0 ny=1 f=1 no=1 out=    15
zr=0 ng=0
 400ns x=     9 y=    15 zx=1 nx=0 zy=1 ny=0 f=0 no=0 out=     0
zr=1 ng=0
 410ns x=     9 y=    15 zx=1 nx=0 zy=1 ny=0 f=0 no=1 out=    -1
zr=0 ng=1
 420ns x=     9 y=    15 zx=1 nx=0 zy=1 ny=0 f=1 no=0 out=     0
zr=1 ng=0
```

```
 430ns x=     9 y=    15 zx=1 nx=0 zy=1 ny=0 f=1 no=1 out=    -1
zr=0 ng=1
 440ns x=     9 y=    15 zx=1 nx=0 zy=1 ny=1 f=0 no=0 out=     0
zr=1 ng=0
 450ns x=     9 y=    15 zx=1 nx=0 zy=1 ny=1 f=0 no=1 out=    -1
zr=0 ng=1
 460ns x=     9 y=    15 zx=1 nx=0 zy=1 ny=1 f=1 no=0 out=    -1
zr=0 ng=1
 470ns x=     9 y=    15 zx=1 nx=0 zy=1 ny=1 f=1 no=1 out=     0
zr=1 ng=0
 480ns x=     9 y=    15 zx=1 nx=1 zy=0 ny=0 f=0 no=0 out=    15
zr=0 ng=0
 490ns x=     9 y=    15 zx=1 nx=1 zy=0 ny=0 f=0 no=1 out=   -16
zr=0 ng=1
 500ns x=     9 y=    15 zx=1 nx=1 zy=0 ny=0 f=1 no=0 out=    14
zr=0 ng=0
```

```
 510ns x=     9 y=    15 zx=1 nx=1 zy=0 ny=0 f=1 no=1 out=   -15
zr=0 ng=1
 520ns x=     9 y=    15 zx=1 nx=1 zy=0 ny=1 f=0 no=0 out=   -16
zr=0 ng=1
 530ns x=     9 y=    15 zx=1 nx=1 zy=0 ny=1 f=0 no=1 out=    15
zr=0 ng=0
 540ns x=     9 y=    15 zx=1 nx=1 zy=0 ny=1 f=1 no=0 out=   -17
zr=0 ng=1
 550ns x=     9 y=    15 zx=1 nx=1 zy=0 ny=1 f=1 no=1 out=    16
zr=0 ng=0
 560ns x=     9 y=    15 zx=1 nx=1 zy=1 ny=0 f=0 no=0 out=     0
zr=1 ng=0
 570ns x=     9 y=    15 zx=1 nx=1 zy=1 ny=0 f=0 no=1 out=    -1
zr=0 ng=1
 580ns x=     9 y=    15 zx=1 nx=1 zy=1 ny=0 f=1 no=0 out=    -1
zr=0 ng=1
```

```
 590ns x=     9 y=    15 zx=1 nx=1 zy=1 ny=0 f=1 no=1 out=     0
zr=1 ng=0
 600ns x=     9 y=    15 zx=1 nx=1 zy=1 ny=1 f=0 no=0 out=    -1
zr=0 ng=1
 610ns x=     9 y=    15 zx=1 nx=1 zy=1 ny=1 f=0 no=1 out=     0
zr=1 ng=0
 620ns x=     9 y=    15 zx=1 nx=1 zy=1 ny=1 f=1 no=0 out=    -2
zr=0 ng=1
 630ns x=     9 y=    15 zx=1 nx=1 zy=1 ny=1 f=1 no=1 out=     1
zr=0 ng=0
 640ns x=     9 y=    15 zx=0 nx=0 zy=0 ny=0 f=0 no=0 out=     9
zr=0 ng=0
```

結語

算術邏輯單元的原理請參考下列文件：

- 少年科技人雜誌 / 2015年6月號 / Nand2Tetris 第二週 -- 布林算術

Nand2Tetris 第三週 -- 自製記憶元件

記憶元件部份包含『單位元記憶、16位元暫存器、程式計數器 PC、RAM、ROM』等等，我們的 verilog 版模組與測試程式如下所示。

程式模組： memory.v

```
/* nand2tetris 的要求應該用下列方式實作，但是由於這樣元件太多會導致 icarus 編譯失敗，
出現下列訊息：

D:\Dropbox\cccweb\db\n2t>iverilog ram_test.v -o ram_test

This application has requested the Runtime to terminate it in an unusual way.
```

```
Please contact the application's support team for more informatio
n.

所以我們在記憶體容量大的時候改用 verilog 的陣列型寫法，這樣就不會
當機了。 */
`include "alu.v"

module DFF (input in, clock, load, output out);
  reg q;
    assign out = q;
  always @(posedge clock) begin
      if (load) q = in;
  end
endmodule

module Bit(input in, clock, load, output out);
```

```
    DFF dff1(in, clock, load, out);
endmodule

module Register(input[15:0] in, input clock, load, output[15:0] o
ut);
  Bit g01(in[15], clock, load, out[15]);
  Bit g02(in[14], clock, load, out[14]);
  Bit g03(in[13], clock, load, out[13]);
  Bit g04(in[12], clock, load, out[12]);
  Bit g05(in[11], clock, load, out[11]);
  Bit g06(in[10], clock, load, out[10]);
  Bit g07(in[9],  clock, load, out[9]);
  Bit g08(in[8],  clock, load, out[8]);
  Bit g09(in[7],  clock, load, out[7]);
  Bit g10(in[6],  clock, load, out[6]);
  Bit g11(in[5],  clock, load, out[5]);
```

```
  Bit g12(in[4],  clock, load, out[4]);
  Bit g13(in[3],  clock, load, out[3]);
  Bit g14(in[2],  clock, load, out[2]);
  Bit g15(in[1],  clock, load, out[1]);
  Bit g16(in[0],  clock, load, out[0]);
endmodule

module PC(input[15:0] in, input clock, load, inc, reset, output[15
:0] out);
  wire[15:0] if1, if2, if3, oInc, o;

  Or g1(load, inc, loadInc);
  Or g2(loadInc, reset, loadIncReset);

  Inc16 inc1(o, oInc);
  And16 g3(o, o, out);
```

```
  Mux16 g4(o,    oInc,   inc,    if1);
  Mux16 g5(if1,  in,     load,   if2);
  Mux16 g6(if2,  16'b0,  reset,  if3);

  Register reg1(if3, clock, loadIncReset, o);
endmodule

module RAM8(input[15:0] in, input clock, load, input[2:0] address
, output[15:0] out);
  wire[15:0] o0,o1,o2,o3,o4,o5,o6,o7;

  DMux8Way g0(1'b1, address, E0, E1, E2, E3, E4, E5, E6, E7);

  And a0(load, E0, L0); Register r0(in, clock, L0, o0);
  And a1(load, E1, L1); Register r1(in, clock, L1, o1);
```

```
  And a2(load, E2, L2); Register r2(in, clock, L2, o2);
  And a3(load, E3, L3); Register r3(in, clock, L3, o3);
  And a4(load, E4, L4); Register r4(in, clock, L4, o4);
  And a5(load, E5, L5); Register r5(in, clock, L5, o5);
  And a6(load, E6, L6); Register r6(in, clock, L6, o6);
  And a7(load, E7, L7); Register r7(in, clock, L7, o7);

  Mux8Way16 g1(o0, o1, o2, o3, o4, o5, o6, o7, address, out);
endmodule

module RAM64(input[15:0] in, input clock, load, input[5:0] addres
s, output[15:0] out);
  wire[15:0] o0,o1,o2,o3,o4,o5,o6,o7;

  DMux8Way g0(1'b1, address[5:3], E0, E1, E2, E3, E4, E5, E6, E7)
;
```

```
  And a0(load, E0, L0); RAM8 m0(in,  clock, L0, address[2:0], o0)
;
  And a1(load, E1, L1); RAM8 m1(in,  clock, L1, address[2:0], o1)
;
  And a2(load, E2, L2); RAM8 m2(in,  clock, L2, address[2:0], o2)
;
  And a3(load, E3, L3); RAM8 m3(in,  clock, L3, address[2:0], o3)
;
  And a4(load, E4, L4); RAM8 m4(in,  clock, L4, address[2:0], o4)
;
  And a5(load, E5, L5); RAM8 m5(in,  clock, L5, address[2:0], o5)
;
  And a6(load, E6, L6); RAM8 m6(in,  clock, L6, address[2:0], o6)
;
  And a7(load, E7, L7); RAM8 m7(in,  clock, L7, address[2:0], o7)
```

```
;

  Mux8Way16 g1(o0, o1, o2, o3, o4, o5, o6, o7, address[5:3], out)
;
endmodule

module ROM32K(input[14:0] address, output[15:0] out);
  reg[15:0] m[0:2**14-1];

  assign out = m[address];
endmodule

module RAM8K(input[15:0] in, input clock, load, input[12:0] addre
ss, output[15:0] out);
  reg[15:0] m[0:2**12-1];
```

```
  assign out = m[address];

  always @(posedge clock) begin
    if (load) m[address] = in;
  end
endmodule

module RAM16K(input[15:0] in, input clock, load, input[13:0] addr
ess, output[15:0] out);
  reg[15:0] m[0:2**13-1];

  assign out = m[address];

  always @(posedge clock) begin
    if (load) m[address] = in;
  end
```

```
endmodule
```

測試程式： ram_test.v

```
`include "memory.v"

module main;
reg[15:0] in;
reg       load, clock;
reg[13:0] address;
wire[15:0] out;

RAM16K m(in, clock, load, address, out);

initial
begin
```

```
  clock=0;
  $monitor("%4dns in=%d clock=%d load=%d address=%d out=%d", $sti
me, in, clock, load, address, out);
    #10 in=3; load=1; address=5;
    #10 load=0;
    #10 $finish;
end

always #2 begin
  clock=~clock;
end

endmodule
```

測試結果

```
D:\Dropbox\cccweb\db\n2t>iverilog ram_test.v -o ram_test

D:\Dropbox\cccweb\db\n2t>vvp ram_test
   0ns in=    x clock=0 load=x address=    x out=    x
   2ns in=    x clock=1 load=x address=    x out=    x
   4ns in=    x clock=0 load=x address=    x out=    x
   6ns in=    x clock=1 load=x address=    x out=    x
   8ns in=    x clock=0 load=x address=    x out=    x
  10ns in=    3 clock=1 load=1 address=    5 out=    3
  12ns in=    3 clock=0 load=1 address=    5 out=    3
  14ns in=    3 clock=1 load=1 address=    5 out=    3
  16ns in=    3 clock=0 load=1 address=    5 out=    3
  18ns in=    3 clock=1 load=1 address=    5 out=    3
  20ns in=    3 clock=0 load=0 address=    5 out=    3
  22ns in=    3 clock=1 load=0 address=    5 out=    3
```

```
  24ns in=    3 clock=0 load=0 address=    5 out=    3
  26ns in=    3 clock=1 load=0 address=    5 out=    3
  28ns in=    3 clock=0 load=0 address=    5 out=    3
  30ns in=    3 clock=1 load=0 address=    5 out=    3
```

另外我們針對 PC 這個元件也單獨進行測試，測試程式與結果如下。

測試程式： PC_test.v

```
`include "memory.v"

module main;
reg[15:0] in;
reg load, inc, reset, clock;
wire[15:0] out;

PC pc(in, clock, load, inc, reset, out);
```

```
initial
begin
  clock = 0;
  $monitor("%4dns clock=%d in=%d reset=%d inc=%d load=%d out=%d",
$stime, clock, in, reset, inc, load, out);
    inc = 0; load = 0; reset=0; in=7;
    #10 reset=1; inc=1;
    #10 reset=0;
    #10 reset=0;
    #30 inc = 0; load=1;
    #30 load = 0; inc=1;
    #30 $finish;
end

always #2 begin
```

```
  clock = clock + 1;
end
```

測試結果

```
D:\Dropbox\cccweb\db\n2t>iverilog pc_test.v -o pc_test

D:\Dropbox\cccweb\db\n2t>vvp pc_test
   0ns clock=0 in=    7 reset=0 inc=0 load=0 out=    x
   2ns clock=1 in=    7 reset=0 inc=0 load=0 out=    x
   4ns clock=0 in=    7 reset=0 inc=0 load=0 out=    x
   6ns clock=1 in=    7 reset=0 inc=0 load=0 out=    x
   8ns clock=0 in=    7 reset=0 inc=0 load=0 out=    x
  10ns clock=1 in=    7 reset=1 inc=1 load=0 out=    x
  12ns clock=0 in=    7 reset=1 inc=1 load=0 out=    x
  14ns clock=1 in=    7 reset=1 inc=1 load=0 out=    0
```

```
16ns clock=0 in=    7 reset=1 inc=1 load=0 out=    0
18ns clock=1 in=    7 reset=1 inc=1 load=0 out=    0
20ns clock=0 in=    7 reset=0 inc=1 load=0 out=    0
22ns clock=1 in=    7 reset=0 inc=1 load=0 out=    1
24ns clock=0 in=    7 reset=0 inc=1 load=0 out=    1
26ns clock=1 in=    7 reset=0 inc=1 load=0 out=    2
28ns clock=0 in=    7 reset=0 inc=1 load=0 out=    2
30ns clock=1 in=    7 reset=0 inc=1 load=0 out=    3
32ns clock=0 in=    7 reset=0 inc=1 load=0 out=    3
34ns clock=1 in=    7 reset=0 inc=1 load=0 out=    4
36ns clock=0 in=    7 reset=0 inc=1 load=0 out=    4
38ns clock=1 in=    7 reset=0 inc=1 load=0 out=    5
40ns clock=0 in=    7 reset=0 inc=1 load=0 out=    5
42ns clock=1 in=    7 reset=0 inc=1 load=0 out=    6
44ns clock=0 in=    7 reset=0 inc=1 load=0 out=    6
46ns clock=1 in=    7 reset=0 inc=1 load=0 out=    7
```

```
48ns clock=0 in=    7 reset=0 inc=1 load=0 out=    7
50ns clock=1 in=    7 reset=0 inc=1 load=0 out=    8
52ns clock=0 in=    7 reset=0 inc=1 load=0 out=    8
54ns clock=1 in=    7 reset=0 inc=1 load=0 out=    9
56ns clock=0 in=    7 reset=0 inc=1 load=0 out=    9
58ns clock=1 in=    7 reset=0 inc=1 load=0 out=   10
60ns clock=0 in=    7 reset=0 inc=0 load=1 out=   10
62ns clock=1 in=    7 reset=0 inc=0 load=1 out=    7
64ns clock=0 in=    7 reset=0 inc=0 load=1 out=    7
66ns clock=1 in=    7 reset=0 inc=0 load=1 out=    7
68ns clock=0 in=    7 reset=0 inc=0 load=1 out=    7
70ns clock=1 in=    7 reset=0 inc=0 load=1 out=    7
72ns clock=0 in=    7 reset=0 inc=0 load=1 out=    7
74ns clock=1 in=    7 reset=0 inc=0 load=1 out=    7
76ns clock=0 in=    7 reset=0 inc=0 load=1 out=    7
78ns clock=1 in=    7 reset=0 inc=0 load=1 out=    7
```

```
 80ns clock=0 in=    7 reset=0 inc=0 load=1 out=    7
 82ns clock=1 in=    7 reset=0 inc=0 load=1 out=    7
 84ns clock=0 in=    7 reset=0 inc=0 load=1 out=    7
 86ns clock=1 in=    7 reset=0 inc=0 load=1 out=    7
 88ns clock=0 in=    7 reset=0 inc=0 load=1 out=    7
 90ns clock=1 in=    7 reset=0 inc=1 load=0 out=    7
 92ns clock=0 in=    7 reset=0 inc=1 load=0 out=    7
 94ns clock=1 in=    7 reset=0 inc=1 load=0 out=    8
 96ns clock=0 in=    7 reset=0 inc=1 load=0 out=    8
 98ns clock=1 in=    7 reset=0 inc=1 load=0 out=    9
100ns clock=0 in=    7 reset=0 inc=1 load=0 out=    9
102ns clock=1 in=    7 reset=0 inc=1 load=0 out=   10
104ns clock=0 in=    7 reset=0 inc=1 load=0 out=   10
106ns clock=1 in=    7 reset=0 inc=1 load=0 out=   11
108ns clock=0 in=    7 reset=0 inc=1 load=0 out=   11
110ns clock=1 in=    7 reset=0 inc=1 load=0 out=   12
```

```
112ns clock=0 in=    7 reset=0 inc=1 load=0 out=   12
114ns clock=1 in=    7 reset=0 inc=1 load=0 out=   13
116ns clock=0 in=    7 reset=0 inc=1 load=0 out=   13
118ns clock=1 in=    7 reset=0 inc=1 load=0 out=   14
120ns clock=0 in=    7 reset=0 inc=1 load=0 out=   14
```

結語

記憶單元的原理請參考下列文件：

- 少年科技人雜誌 / 2015年6月號 / Nand2Tetris 第 3 週 -- 記憶體

Nand2Tetris 第四週 -- 學習機器語言

關於第四週的學習機器語言部份，是要我們設計 HackComputer 的兩個組合語言程式，一個是控制鍵盤與畫面的 Fill.asm ，另一個是計算乘法的 Mult.asm。

為了避免公佈答案，在此我只寫下我在這兩個程式所採用的算法虛擬碼，以下是 Fill.asm 的高階虛擬碼。

```
forever
  arr = SCREEN
  for (i=0; i<8192; i++) {
    if (*KBD != 0)
      arr[i] = -1
    else
      arr[i] = 0
  }
  goto forever;
```

以下是較接近組合語言的低階虛擬碼，此虛擬和上面的並不完全一致，因為我有進一步簡化過。

```
  arr = SCREEN
  n=8192
  i=0
```

```
FOREVER:
LOOP:
  if (i==n) goto ENDLOOP
  if (*KBD != 0)
    arr[i] = 0;  // 0x0000
  else
    arr[i] = -1; // 0xFFFF
  i++;
ENDLOOP:
  goto FOREVER
```

組合語言 : Mult.asm

```
// Multiplies R0 and R1 and stores the result in R2.
// (R0, R1, R2 refer to RAM[0], RAM[1], and RAM[2], respectively.
)
```

```
// Put your code here.
  a = 0
LOOP:
  if (a <= 0) goto EXIT
    a=a-1;
  R2 = R2 + R1;
  goto LOOP
```

在寫好組合語言程式 Fill.asm 與 Mult.asm 之後，您可以使用 nand2tetris 所提供的 CPUEmulator 來執行並驗證您寫出來的組合語言程式。

結語

組合語言單元的原理請參考下列文件：

- 少年科技人雜誌 / 2015年6月號 / Nand2Tetris 第 4 週 -- 機器語言

Nand2Tetris 第五週 -- 自製處理器與電腦

本週的作業是設計處理器 HackCPU 與電腦 HackComputer，其中還包含了可以與週邊連接的記憶體部份。Verilog 版的設計如下:

程式模組： computer.v

```
`include "memory.v"

module Memory(input[15:0] in, input clock, load, input[14:0] addr
ess, output[15:0] out);
  wire[15:0] outM, outS, outK, outSK;

  Not g1(address[14], N14);
  And g2(N14, load, Mload);
  And g3(address[14], load, Sload);
```

```
  RAM16K ram16k(in, clock, Mload, address[13:0], outM);
  RAM8K  screen(in, clock, Sload, address[12:0], outS);
  Register keyboard(16'h0F0F, clock, 1'b0, outK);

  Mux16 g4(outM, outSK, address[14], out);
  Mux16 g5(outS, outK,  address[13], outSK);
endmodule

module CPU(input[15:0] inM, I, input clock, reset, output[15:0] o
utM, output writeM, output[14:0] addressM, pc);
  wire[15:0] Ain, Aout, AorM, ALUout, Dout, pcOut, addressMOut;
  Or  g1(ng, zr, ngzr);    // ngzr = (ng|zr)
  Not g2(ngzr, g);      // g = out > 0 = !(ng|zr);  ng = out < 0;
 zr = out = 0
  And g3(ng, I[2], passLT);     // ngLT = (ng&LT)
  And g4(zr, I[1], passEQ);     // zrEQ = (zr&EQ)
```

```
  And g5(g,  I[0], passGT);    // gGT = (g&GT)
  Or  g6(passLT, passEQ, passLE);
  Or  g7(passLE, passGT, pass);

  And g8(I[15], pass, PCload);     // PCload = I15&J

  // ALU
  Mux16 g9(Aout, inM, I[12], AorM); // Mux ALU in : cAorM = I[12]

  ALU alu(Dout, AorM, I[11], I[10], I[9], I[8], I[7], I[6], ALUou
t, zr, ng);

  PC pc1(Aout, clock, PCload, 1'b1, reset, pcOut);
    assign pc = pcOut[14:0];

  // A register
```

```
  Not g10(I[15], Atype);
  And g11(I[15], I[5], AluToA); // AluToA = I[15]&d1
  Or  g12(Atype, AluToA, Aload);

  Mux16 g13(I, ALUout, AluToA, Ain); // sel=I[15]
  Register A(Ain, clock, Aload, Aout);

  // D register
  And g14(I[15], I[4], Dload); // Aload = I[15]&d2
  Register D(ALUout, clock, Dload, Dout);

  // output
    assign addressM = Aout[14:0];
  And g16(I[15], I[3], writeM); // writeM = I[15] & d3
  And16 g17(ALUout, ALUout, outM);
endmodule
```

```
module Computer(input clock, reset);
  wire[15:0] inM, outM, I;
    wire[14:0] addressM, pc;

  Memory ram(inM, !clock, loadM, addressM, outM);
  ROM32K rom(pc, I);
  CPU    cpu(outM, I, clock, reset, inM, loadM, addressM, pc);
endmodule
```

測試程式： computer_test.v

```
`include "computer.v"

module main;
reg reset, clock;
```

```
Computer c(clock, reset);

integer i;

initial
begin
  $readmemb("sum.hack", c.rom.m);
  for (i=0; i < 32; i=i+1) begin
    $display("%4x: %x", i, c.rom.m[i]);
  end
  $monitor("%4dns clock=%d pc=%d I=%d A=%d D=%d M=%d", $stime, cl
ock, c.pc, c.I, c.addressM, c.cpu.Dout, c.outM);
    clock = 0;
    #10 reset=1;
    #30 reset=0;
```

```
end

always #5 begin
  clock = clock + 1;
end

initial #1800 $finish;

endmodule
```

輸入檔 : sum.hack

```
0000000000010000  //    @i // i refers to some RAM location
1110111111001000  //    M=1 // i=1
0000000000010001  //    @sum // sum refers to some RAM location
1110101010001000  //    M=0 // sum=0
```

```
0000000000010000  // (LOOP) @i
1111110000010000  //    D=M // D = i
0000000000001010  //    @10
1110010011010000  //    D=D-A // D = i - 10
0000000000010010  //    @END
1110001100000001  //    D;JGT // If (i-100) > 0 goto END
0000000000010000  //    @i
1111110000010000  //    D=M // D = i
0000000000010001  //    @sum
1111000010001000  //    M=D+M // sum += i
0000000000010000  //    @i
1111110111001000  //    M=M+1 // i++
0000000000000100  //    @LOOP
1110101010000111  //    0;JMP // Got LOOP
0000000000010010  // (END)  @END
1110101010000111  //    0;JMP // Infinite loop
```

測試結果

```
WARNING: computer_test.v:12: $readmemb(sum.hack): Not enough word
s in the file for the requested range [0:16383].
00000000: 0010
00000001: efc8
00000002: 0011
00000003: ea88
00000004: 0010
00000005: fc10
00000006: 000a
00000007: e4d0
00000008: 0012
00000009: e301
0000000a: 0010
0000000b: fc10
```

```
0000000c: 0011
0000000d: f088
0000000e: 0010
0000000f: fdc8
00000010: 0004
00000011: ea87
00000012: 0012
00000013: ea87
00000014: xxxx
00000015: xxxx
00000016: xxxx
00000017: xxxx
00000018: xxxx
00000019: xxxx
0000001a: xxxx
0000001b: xxxx
```

```
0000001c: xxxx
0000001d: xxxx
0000001e: xxxx
0000001f: xxxx
   0ns clock=0 pc=    x I=    x A=    x D=    x M=    x
   5ns clock=1 pc=    x I=    x A=    x D=    x M=    x
  10ns clock=0 pc=    x I=    x A=    x D=    x M=    x
  15ns clock=1 pc=    0 I=   16 A=    x D=    x M=    x
  20ns clock=0 pc=    0 I=   16 A=    x D=    x M=    x
  25ns clock=1 pc=    0 I=   16 A=   16 D=    x M=    x
  30ns clock=0 pc=    0 I=   16 A=   16 D=    x M=    x
  35ns clock=1 pc=    0 I=   16 A=   16 D=    x M=    x
  40ns clock=0 pc=    0 I=   16 A=   16 D=    x M=    x
  45ns clock=1 pc=    1 I=61384 A=   16 D=    x M=    x
  50ns clock=0 pc=    1 I=61384 A=   16 D=    x M=    1
  55ns clock=1 pc=    2 I=   17 A=   16 D=    x M=    1
```

```
 60ns clock=0 pc=    2 I=   17 A=   16 D=    x M=    1
 65ns clock=1 pc=    3 I=60040 A=   17 D=    x M=    x
 70ns clock=0 pc=    3 I=60040 A=   17 D=    x M=    0
 75ns clock=1 pc=    4 I=   16 A=   17 D=    x M=    0
 80ns clock=0 pc=    4 I=   16 A=   17 D=    x M=    0
 85ns clock=1 pc=    5 I=64528 A=   16 D=    x M=    1
 90ns clock=0 pc=    5 I=64528 A=   16 D=    x M=    1
 95ns clock=1 pc=    6 I=   10 A=   16 D=    1 M=    1
100ns clock=0 pc=    6 I=   10 A=   16 D=    1 M=    1
105ns clock=1 pc=    7 I=58576 A=   10 D=    1 M=    x
110ns clock=0 pc=    7 I=58576 A=   10 D=    1 M=    x
115ns clock=1 pc=    8 I=   18 A=   10 D=65527 M=    x
120ns clock=0 pc=    8 I=   18 A=   10 D=65527 M=    x
125ns clock=1 pc=    9 I=58113 A=   18 D=65527 M=    x
130ns clock=0 pc=    9 I=58113 A=   18 D=65527 M=    x
135ns clock=1 pc=   10 I=   16 A=   18 D=65527 M=    x
```

```
140ns clock=0 pc=   10 I=   16 A=   18 D=65527 M=    x
145ns clock=1 pc=   11 I=64528 A=   16 D=65527 M=    1
150ns clock=0 pc=   11 I=64528 A=   16 D=65527 M=    1
155ns clock=1 pc=   12 I=   17 A=   16 D=    1 M=    1
160ns clock=0 pc=   12 I=   17 A=   16 D=    1 M=    1
165ns clock=1 pc=   13 I=61576 A=   17 D=    1 M=    0
170ns clock=0 pc=   13 I=61576 A=   17 D=    1 M=    1
175ns clock=1 pc=   14 I=   16 A=   17 D=    1 M=    1
180ns clock=0 pc=   14 I=   16 A=   17 D=    1 M=    1
185ns clock=1 pc=   15 I=64968 A=   16 D=    1 M=    1
190ns clock=0 pc=   15 I=64968 A=   16 D=    1 M=    2
195ns clock=1 pc=   16 I=    4 A=   16 D=    1 M=    2
200ns clock=0 pc=   16 I=    4 A=   16 D=    1 M=    2
205ns clock=1 pc=   17 I=60039 A=    4 D=    1 M=    x
210ns clock=0 pc=   17 I=60039 A=    4 D=    1 M=    x
215ns clock=1 pc=    4 I=   16 A=    4 D=    1 M=    x
```

```
220ns clock=0 pc=    4 I=   16 A=    4 D=    1 M=    x
225ns clock=1 pc=    5 I=64528 A=   16 D=    1 M=    2
230ns clock=0 pc=    5 I=64528 A=   16 D=    1 M=    2
235ns clock=1 pc=    6 I=   10 A=   16 D=    2 M=    2
240ns clock=0 pc=    6 I=   10 A=   16 D=    2 M=    2
245ns clock=1 pc=    7 I=58576 A=   10 D=    2 M=    x
250ns clock=0 pc=    7 I=58576 A=   10 D=    2 M=    x
255ns clock=1 pc=    8 I=   18 A=   10 D=65528 M=    x
260ns clock=0 pc=    8 I=   18 A=   10 D=65528 M=    x
265ns clock=1 pc=    9 I=58113 A=   18 D=65528 M=    x
270ns clock=0 pc=    9 I=58113 A=   18 D=65528 M=    x
275ns clock=1 pc=   10 I=   16 A=   18 D=65528 M=    x
280ns clock=0 pc=   10 I=   16 A=   18 D=65528 M=    x
285ns clock=1 pc=   11 I=64528 A=   16 D=65528 M=    2
290ns clock=0 pc=   11 I=64528 A=   16 D=65528 M=    2
295ns clock=1 pc=   12 I=   17 A=   16 D=    2 M=    2
```

```
300ns clock=0 pc=   12 I=   17 A=   16 D=    2 M=    2
305ns clock=1 pc=   13 I=61576 A=   17 D=    2 M=    1
310ns clock=0 pc=   13 I=61576 A=   17 D=    2 M=    3
315ns clock=1 pc=   14 I=   16 A=   17 D=    2 M=    3
320ns clock=0 pc=   14 I=   16 A=   17 D=    2 M=    3
325ns clock=1 pc=   15 I=64968 A=   16 D=    2 M=    2
330ns clock=0 pc=   15 I=64968 A=   16 D=    2 M=    3
335ns clock=1 pc=   16 I=    4 A=   16 D=    2 M=    3
340ns clock=0 pc=   16 I=    4 A=   16 D=    2 M=    3
345ns clock=1 pc=   17 I=60039 A=    4 D=    2 M=    x
350ns clock=0 pc=   17 I=60039 A=    4 D=    2 M=    x
355ns clock=1 pc=    4 I=   16 A=    4 D=    2 M=    x
360ns clock=0 pc=    4 I=   16 A=    4 D=    2 M=    x
365ns clock=1 pc=    5 I=64528 A=   16 D=    2 M=    3
370ns clock=0 pc=    5 I=64528 A=   16 D=    2 M=    3
375ns clock=1 pc=    6 I=   10 A=   16 D=    3 M=    3
```

```
380ns clock=0 pc=    6 I=   10 A=   16 D=    3 M=    3
385ns clock=1 pc=    7 I=58576 A=   10 D=    3 M=    x
390ns clock=0 pc=    7 I=58576 A=   10 D=    3 M=    x
395ns clock=1 pc=    8 I=   18 A=   10 D=65529 M=    x
400ns clock=0 pc=    8 I=   18 A=   10 D=65529 M=    x
405ns clock=1 pc=    9 I=58113 A=   18 D=65529 M=    x
410ns clock=0 pc=    9 I=58113 A=   18 D=65529 M=    x
415ns clock=1 pc=   10 I=   16 A=   18 D=65529 M=    x
420ns clock=0 pc=   10 I=   16 A=   18 D=65529 M=    x
425ns clock=1 pc=   11 I=64528 A=   16 D=65529 M=    3
430ns clock=0 pc=   11 I=64528 A=   16 D=65529 M=    3
435ns clock=1 pc=   12 I=   17 A=   16 D=    3 M=    3
440ns clock=0 pc=   12 I=   17 A=   16 D=    3 M=    3
445ns clock=1 pc=   13 I=61576 A=   17 D=    3 M=    3
450ns clock=0 pc=   13 I=61576 A=   17 D=    3 M=    6
455ns clock=1 pc=   14 I=   16 A=   17 D=    3 M=    6
```

```
460ns clock=0 pc=   14 I=   16 A=   17 D=    3 M=    6
465ns clock=1 pc=   15 I=64968 A=   16 D=    3 M=    3
470ns clock=0 pc=   15 I=64968 A=   16 D=    3 M=    4
475ns clock=1 pc=   16 I=    4 A=   16 D=    3 M=    4
480ns clock=0 pc=   16 I=    4 A=   16 D=    3 M=    4
485ns clock=1 pc=   17 I=60039 A=    4 D=    3 M=    x
490ns clock=0 pc=   17 I=60039 A=    4 D=    3 M=    x
495ns clock=1 pc=    4 I=   16 A=    4 D=    3 M=    x
500ns clock=0 pc=    4 I=   16 A=    4 D=    3 M=    x
505ns clock=1 pc=    5 I=64528 A=   16 D=    3 M=    4
510ns clock=0 pc=    5 I=64528 A=   16 D=    3 M=    4
515ns clock=1 pc=    6 I=   10 A=   16 D=    4 M=    4
520ns clock=0 pc=    6 I=   10 A=   16 D=    4 M=    4
525ns clock=1 pc=    7 I=58576 A=   10 D=    4 M=    x
530ns clock=0 pc=    7 I=58576 A=   10 D=    4 M=    x
535ns clock=1 pc=    8 I=   18 A=   10 D=65530 M=    x
```

```
540ns clock=0 pc=    8 I=   18 A=   10 D=65530 M=    x
545ns clock=1 pc=    9 I=58113 A=   18 D=65530 M=    x
550ns clock=0 pc=    9 I=58113 A=   18 D=65530 M=    x
555ns clock=1 pc=   10 I=   16 A=   18 D=65530 M=    x
560ns clock=0 pc=   10 I=   16 A=   18 D=65530 M=    x
565ns clock=1 pc=   11 I=64528 A=   16 D=65530 M=    4
570ns clock=0 pc=   11 I=64528 A=   16 D=65530 M=    4
575ns clock=1 pc=   12 I=   17 A=   16 D=    4 M=    4
580ns clock=0 pc=   12 I=   17 A=   16 D=    4 M=    4
585ns clock=1 pc=   13 I=61576 A=   17 D=    4 M=    6
590ns clock=0 pc=   13 I=61576 A=   17 D=    4 M=   10
595ns clock=1 pc=   14 I=   16 A=   17 D=    4 M=   10
600ns clock=0 pc=   14 I=   16 A=   17 D=    4 M=   10
605ns clock=1 pc=   15 I=64968 A=   16 D=    4 M=    4
610ns clock=0 pc=   15 I=64968 A=   16 D=    4 M=    5
615ns clock=1 pc=   16 I=    4 A=   16 D=    4 M=    5
```

```
620ns clock=0 pc=   16 I=    4 A=   16 D=    4 M=    5
625ns clock=1 pc=   17 I=60039 A=    4 D=    4 M=    x
630ns clock=0 pc=   17 I=60039 A=    4 D=    4 M=    x
635ns clock=1 pc=    4 I=   16 A=    4 D=    4 M=    x
640ns clock=0 pc=    4 I=   16 A=    4 D=    4 M=    x
645ns clock=1 pc=    5 I=64528 A=   16 D=    4 M=    5
650ns clock=0 pc=    5 I=64528 A=   16 D=    4 M=    5
655ns clock=1 pc=    6 I=   10 A=   16 D=    5 M=    5
660ns clock=0 pc=    6 I=   10 A=   16 D=    5 M=    5
665ns clock=1 pc=    7 I=58576 A=   10 D=    5 M=    x
670ns clock=0 pc=    7 I=58576 A=   10 D=    5 M=    x
675ns clock=1 pc=    8 I=   18 A=   10 D=65531 M=    x
680ns clock=0 pc=    8 I=   18 A=   10 D=65531 M=    x
685ns clock=1 pc=    9 I=58113 A=   18 D=65531 M=    x
690ns clock=0 pc=    9 I=58113 A=   18 D=65531 M=    x
695ns clock=1 pc=   10 I=   16 A=   18 D=65531 M=    x
```

```
700ns clock=0 pc=   10 I=   16 A=   18 D=65531 M=    x
705ns clock=1 pc=   11 I=64528 A=   16 D=65531 M=    5
710ns clock=0 pc=   11 I=64528 A=   16 D=65531 M=    5
715ns clock=1 pc=   12 I=   17 A=   16 D=    5 M=    5
720ns clock=0 pc=   12 I=   17 A=   16 D=    5 M=    5
725ns clock=1 pc=   13 I=61576 A=   17 D=    5 M=   10
730ns clock=0 pc=   13 I=61576 A=   17 D=    5 M=   15
735ns clock=1 pc=   14 I=   16 A=   17 D=    5 M=   15
740ns clock=0 pc=   14 I=   16 A=   17 D=    5 M=   15
745ns clock=1 pc=   15 I=64968 A=   16 D=    5 M=    5
750ns clock=0 pc=   15 I=64968 A=   16 D=    5 M=    6
755ns clock=1 pc=   16 I=    4 A=   16 D=    5 M=    6
760ns clock=0 pc=   16 I=    4 A=   16 D=    5 M=    6
765ns clock=1 pc=   17 I=60039 A=    4 D=    5 M=    x
770ns clock=0 pc=   17 I=60039 A=    4 D=    5 M=    x
775ns clock=1 pc=    4 I=   16 A=    4 D=    5 M=    x
```

```
780ns clock=0 pc=    4 I=   16 A=    4 D=    5 M=    x
785ns clock=1 pc=    5 I=64528 A=   16 D=    5 M=    6
790ns clock=0 pc=    5 I=64528 A=   16 D=    5 M=    6
795ns clock=1 pc=    6 I=   10 A=   16 D=    6 M=    6
800ns clock=0 pc=    6 I=   10 A=   16 D=    6 M=    6
805ns clock=1 pc=    7 I=58576 A=   10 D=    6 M=    x
810ns clock=0 pc=    7 I=58576 A=   10 D=    6 M=    x
815ns clock=1 pc=    8 I=   18 A=   10 D=65532 M=    x
820ns clock=0 pc=    8 I=   18 A=   10 D=65532 M=    x
825ns clock=1 pc=    9 I=58113 A=   18 D=65532 M=    x
830ns clock=0 pc=    9 I=58113 A=   18 D=65532 M=    x
835ns clock=1 pc=   10 I=   16 A=   18 D=65532 M=    x
840ns clock=0 pc=   10 I=   16 A=   18 D=65532 M=    x
845ns clock=1 pc=   11 I=64528 A=   16 D=65532 M=    6
850ns clock=0 pc=   11 I=64528 A=   16 D=65532 M=    6
855ns clock=1 pc=   12 I=   17 A=   16 D=    6 M=    6
```

```
860ns clock=0 pc=   12 I=   17 A=   16 D=    6 M=    6
865ns clock=1 pc=   13 I=61576 A=   17 D=    6 M=   15
870ns clock=0 pc=   13 I=61576 A=   17 D=    6 M=   21
875ns clock=1 pc=   14 I=   16 A=   17 D=    6 M=   21
880ns clock=0 pc=   14 I=   16 A=   17 D=    6 M=   21
885ns clock=1 pc=   15 I=64968 A=   16 D=    6 M=    6
890ns clock=0 pc=   15 I=64968 A=   16 D=    6 M=    7
895ns clock=1 pc=   16 I=    4 A=   16 D=    6 M=    7
900ns clock=0 pc=   16 I=    4 A=   16 D=    6 M=    7
905ns clock=1 pc=   17 I=60039 A=    4 D=    6 M=    x
910ns clock=0 pc=   17 I=60039 A=    4 D=    6 M=    x
915ns clock=1 pc=    4 I=   16 A=    4 D=    6 M=    x
920ns clock=0 pc=    4 I=   16 A=    4 D=    6 M=    x
925ns clock=1 pc=    5 I=64528 A=   16 D=    6 M=    7
930ns clock=0 pc=    5 I=64528 A=   16 D=    6 M=    7
935ns clock=1 pc=    6 I=   10 A=   16 D=    7 M=    7
```

```
 940ns clock=0 pc=    6 I=   10 A=   16 D=    7 M=    7
 945ns clock=1 pc=    7 I=58576 A=   10 D=    7 M=    x
 950ns clock=0 pc=    7 I=58576 A=   10 D=    7 M=    x
 955ns clock=1 pc=    8 I=   18 A=   10 D=65533 M=    x
 960ns clock=0 pc=    8 I=   18 A=   10 D=65533 M=    x
 965ns clock=1 pc=    9 I=58113 A=   18 D=65533 M=    x
 970ns clock=0 pc=    9 I=58113 A=   18 D=65533 M=    x
 975ns clock=1 pc=   10 I=   16 A=   18 D=65533 M=    x
 980ns clock=0 pc=   10 I=   16 A=   18 D=65533 M=    x
 985ns clock=1 pc=   11 I=64528 A=   16 D=65533 M=    7
 990ns clock=0 pc=   11 I=64528 A=   16 D=65533 M=    7
 995ns clock=1 pc=   12 I=   17 A=   16 D=    7 M=    7
1000ns clock=0 pc=   12 I=   17 A=   16 D=    7 M=    7
1005ns clock=1 pc=   13 I=61576 A=   17 D=    7 M=   21
1010ns clock=0 pc=   13 I=61576 A=   17 D=    7 M=   28
1015ns clock=1 pc=   14 I=   16 A=   17 D=    7 M=   28
```

```
1020ns clock=0 pc=   14 I=   16 A=   17 D=    7 M=   28
1025ns clock=1 pc=   15 I=64968 A=   16 D=    7 M=    7
1030ns clock=0 pc=   15 I=64968 A=   16 D=    7 M=    8
1035ns clock=1 pc=   16 I=    4 A=   16 D=    7 M=    8
1040ns clock=0 pc=   16 I=    4 A=   16 D=    7 M=    8
1045ns clock=1 pc=   17 I=60039 A=    4 D=    7 M=    x
1050ns clock=0 pc=   17 I=60039 A=    4 D=    7 M=    x
1055ns clock=1 pc=    4 I=   16 A=    4 D=    7 M=    x
1060ns clock=0 pc=    4 I=   16 A=    4 D=    7 M=    x
1065ns clock=1 pc=    5 I=64528 A=   16 D=    7 M=    8
1070ns clock=0 pc=    5 I=64528 A=   16 D=    7 M=    8
1075ns clock=1 pc=    6 I=   10 A=   16 D=    8 M=    8
1080ns clock=0 pc=    6 I=   10 A=   16 D=    8 M=    8
1085ns clock=1 pc=    7 I=58576 A=   10 D=    8 M=    x
1090ns clock=0 pc=    7 I=58576 A=   10 D=    8 M=    x
1095ns clock=1 pc=    8 I=   18 A=   10 D=65534 M=    x
```

```
1100ns clock=0 pc=    8 I=   18 A=   10 D=65534 M=    x
1105ns clock=1 pc=    9 I=58113 A=   18 D=65534 M=    x
1110ns clock=0 pc=    9 I=58113 A=   18 D=65534 M=    x
1115ns clock=1 pc=   10 I=   16 A=   18 D=65534 M=    x
1120ns clock=0 pc=   10 I=   16 A=   18 D=65534 M=    x
1125ns clock=1 pc=   11 I=64528 A=   16 D=65534 M=    8
1130ns clock=0 pc=   11 I=64528 A=   16 D=65534 M=    8
1135ns clock=1 pc=   12 I=   17 A=   16 D=    8 M=    8
1140ns clock=0 pc=   12 I=   17 A=   16 D=    8 M=    8
1145ns clock=1 pc=   13 I=61576 A=   17 D=    8 M=   28
1150ns clock=0 pc=   13 I=61576 A=   17 D=    8 M=   36
1155ns clock=1 pc=   14 I=   16 A=   17 D=    8 M=   36
1160ns clock=0 pc=   14 I=   16 A=   17 D=    8 M=   36
1165ns clock=1 pc=   15 I=64968 A=   16 D=    8 M=    8
1170ns clock=0 pc=   15 I=64968 A=   16 D=    8 M=    9
1175ns clock=1 pc=   16 I=    4 A=   16 D=    8 M=    9
```

```
1180ns clock=0 pc=   16 I=    4 A=   16 D=    8 M=    9
1185ns clock=1 pc=   17 I=60039 A=    4 D=    8 M=    x
1190ns clock=0 pc=   17 I=60039 A=    4 D=    8 M=    x
1195ns clock=1 pc=    4 I=   16 A=    4 D=    8 M=    x
1200ns clock=0 pc=    4 I=   16 A=    4 D=    8 M=    x
1205ns clock=1 pc=    5 I=64528 A=   16 D=    8 M=    9
1210ns clock=0 pc=    5 I=64528 A=   16 D=    8 M=    9
1215ns clock=1 pc=    6 I=   10 A=   16 D=    9 M=    9
1220ns clock=0 pc=    6 I=   10 A=   16 D=    9 M=    9
1225ns clock=1 pc=    7 I=58576 A=   10 D=    9 M=    x
1230ns clock=0 pc=    7 I=58576 A=   10 D=    9 M=    x
1235ns clock=1 pc=    8 I=   18 A=   10 D=65535 M=    x
1240ns clock=0 pc=    8 I=   18 A=   10 D=65535 M=    x
1245ns clock=1 pc=    9 I=58113 A=   18 D=65535 M=    x
1250ns clock=0 pc=    9 I=58113 A=   18 D=65535 M=    x
1255ns clock=1 pc=   10 I=   16 A=   18 D=65535 M=    x
```

```
1260ns clock=0 pc=   10 I=   16 A=   18 D=65535 M=    x
1265ns clock=1 pc=   11 I=64528 A=   16 D=65535 M=    9
1270ns clock=0 pc=   11 I=64528 A=   16 D=65535 M=    9
1275ns clock=1 pc=   12 I=   17 A=   16 D=    9 M=    9
1280ns clock=0 pc=   12 I=   17 A=   16 D=    9 M=    9
1285ns clock=1 pc=   13 I=61576 A=   17 D=    9 M=   36
1290ns clock=0 pc=   13 I=61576 A=   17 D=    9 M=   45
1295ns clock=1 pc=   14 I=   16 A=   17 D=    9 M=   45
1300ns clock=0 pc=   14 I=   16 A=   17 D=    9 M=   45
1305ns clock=1 pc=   15 I=64968 A=   16 D=    9 M=    9
1310ns clock=0 pc=   15 I=64968 A=   16 D=    9 M=   10
1315ns clock=1 pc=   16 I=    4 A=   16 D=    9 M=   10
1320ns clock=0 pc=   16 I=    4 A=   16 D=    9 M=   10
1325ns clock=1 pc=   17 I=60039 A=    4 D=    9 M=    x
1330ns clock=0 pc=   17 I=60039 A=    4 D=    9 M=    x
1335ns clock=1 pc=    4 I=   16 A=    4 D=    9 M=    x
```

```
1340ns clock=0 pc=    4 I=   16 A=    4 D=    9 M=    x
1345ns clock=1 pc=    5 I=64528 A=   16 D=    9 M=   10
1350ns clock=0 pc=    5 I=64528 A=   16 D=    9 M=   10
1355ns clock=1 pc=    6 I=   10 A=   16 D=   10 M=   10
1360ns clock=0 pc=    6 I=   10 A=   16 D=   10 M=   10
1365ns clock=1 pc=    7 I=58576 A=   10 D=   10 M=    x
1370ns clock=0 pc=    7 I=58576 A=   10 D=   10 M=    x
1375ns clock=1 pc=    8 I=   18 A=   10 D=    0 M=    x
1380ns clock=0 pc=    8 I=   18 A=   10 D=    0 M=    x
1385ns clock=1 pc=    9 I=58113 A=   18 D=    0 M=    x
1390ns clock=0 pc=    9 I=58113 A=   18 D=    0 M=    x
1395ns clock=1 pc=   10 I=   16 A=   18 D=    0 M=    x
1400ns clock=0 pc=   10 I=   16 A=   18 D=    0 M=    x
1405ns clock=1 pc=   11 I=64528 A=   16 D=    0 M=   10
1410ns clock=0 pc=   11 I=64528 A=   16 D=    0 M=   10
1415ns clock=1 pc=   12 I=   17 A=   16 D=   10 M=   10
```

```
1420ns clock=0 pc=   12 I=   17 A=   16 D=   10 M=   10
1425ns clock=1 pc=   13 I=61576 A=   17 D=   10 M=   45
1430ns clock=0 pc=   13 I=61576 A=   17 D=   10 M=   55
1435ns clock=1 pc=   14 I=   16 A=   17 D=   10 M=   55
1440ns clock=0 pc=   14 I=   16 A=   17 D=   10 M=   55
1445ns clock=1 pc=   15 I=64968 A=   16 D=   10 M=   10
1450ns clock=0 pc=   15 I=64968 A=   16 D=   10 M=   11
1455ns clock=1 pc=   16 I=    4 A=   16 D=   10 M=   11
1460ns clock=0 pc=   16 I=    4 A=   16 D=   10 M=   11
1465ns clock=1 pc=   17 I=60039 A=    4 D=   10 M=    x
1470ns clock=0 pc=   17 I=60039 A=    4 D=   10 M=    x
1475ns clock=1 pc=    4 I=   16 A=    4 D=   10 M=    x
1480ns clock=0 pc=    4 I=   16 A=    4 D=   10 M=    x
1485ns clock=1 pc=    5 I=64528 A=   16 D=   10 M=   11
1490ns clock=0 pc=    5 I=64528 A=   16 D=   10 M=   11
1495ns clock=1 pc=    6 I=   10 A=   16 D=   11 M=   11
```

```
1500ns clock=0 pc=    6 I=   10 A=   16 D=   11 M=   11
1505ns clock=1 pc=    7 I=58576 A=   10 D=   11 M=    x
1510ns clock=0 pc=    7 I=58576 A=   10 D=   11 M=    x
1515ns clock=1 pc=    8 I=   18 A=   10 D=    1 M=    x
1520ns clock=0 pc=    8 I=   18 A=   10 D=    1 M=    x
1525ns clock=1 pc=    9 I=58113 A=   18 D=    1 M=    x
1530ns clock=0 pc=    9 I=58113 A=   18 D=    1 M=    x
1535ns clock=1 pc=   18 I=   18 A=   18 D=    1 M=    x
1540ns clock=0 pc=   18 I=   18 A=   18 D=    1 M=    x
1545ns clock=1 pc=   19 I=60039 A=   18 D=    1 M=    x
1550ns clock=0 pc=   19 I=60039 A=   18 D=    1 M=    x
1555ns clock=1 pc=   18 I=   18 A=   18 D=    1 M=    x
1560ns clock=0 pc=   18 I=   18 A=   18 D=    1 M=    x
1565ns clock=1 pc=   19 I=60039 A=   18 D=    1 M=    x
1570ns clock=0 pc=   19 I=60039 A=   18 D=    1 M=    x
1575ns clock=1 pc=   18 I=   18 A=   18 D=    1 M=    x
```

```
1580ns clock=0 pc=   18 I=   18 A=   18 D=    1 M=    x
1585ns clock=1 pc=   19 I=60039 A=   18 D=    1 M=    x
1590ns clock=0 pc=   19 I=60039 A=   18 D=    1 M=    x
1595ns clock=1 pc=   18 I=   18 A=   18 D=    1 M=    x
1600ns clock=0 pc=   18 I=   18 A=   18 D=    1 M=    x
1605ns clock=1 pc=   19 I=60039 A=   18 D=    1 M=    x
1610ns clock=0 pc=   19 I=60039 A=   18 D=    1 M=    x
1615ns clock=1 pc=   18 I=   18 A=   18 D=    1 M=    x
1620ns clock=0 pc=   18 I=   18 A=   18 D=    1 M=    x
1625ns clock=1 pc=   19 I=60039 A=   18 D=    1 M=    x
1630ns clock=0 pc=   19 I=60039 A=   18 D=    1 M=    x
1635ns clock=1 pc=   18 I=   18 A=   18 D=    1 M=    x
1640ns clock=0 pc=   18 I=   18 A=   18 D=    1 M=    x
1645ns clock=1 pc=   19 I=60039 A=   18 D=    1 M=    x
1650ns clock=0 pc=   19 I=60039 A=   18 D=    1 M=    x
1655ns clock=1 pc=   18 I=   18 A=   18 D=    1 M=    x
```

```
1660ns clock=0 pc=   18 I=   18 A=   18 D=    1 M=    x
1665ns clock=1 pc=   19 I=60039 A=   18 D=    1 M=    x
1670ns clock=0 pc=   19 I=60039 A=   18 D=    1 M=    x
1675ns clock=1 pc=   18 I=   18 A=   18 D=    1 M=    x
1680ns clock=0 pc=   18 I=   18 A=   18 D=    1 M=    x
1685ns clock=1 pc=   19 I=60039 A=   18 D=    1 M=    x
1690ns clock=0 pc=   19 I=60039 A=   18 D=    1 M=    x
1695ns clock=1 pc=   18 I=   18 A=   18 D=    1 M=    x
1700ns clock=0 pc=   18 I=   18 A=   18 D=    1 M=    x
1705ns clock=1 pc=   19 I=60039 A=   18 D=    1 M=    x
1710ns clock=0 pc=   19 I=60039 A=   18 D=    1 M=    x
1715ns clock=1 pc=   18 I=   18 A=   18 D=    1 M=    x
1720ns clock=0 pc=   18 I=   18 A=   18 D=    1 M=    x
1725ns clock=1 pc=   19 I=60039 A=   18 D=    1 M=    x
1730ns clock=0 pc=   19 I=60039 A=   18 D=    1 M=    x
1735ns clock=1 pc=   18 I=   18 A=   18 D=    1 M=    x
```

```
1740ns clock=0 pc=   18 I=   18 A=   18 D=    1 M=    x
1745ns clock=1 pc=   19 I=60039 A=   18 D=    1 M=    x
1750ns clock=0 pc=   19 I=60039 A=   18 D=    1 M=    x
1755ns clock=1 pc=   18 I=   18 A=   18 D=    1 M=    x
1760ns clock=0 pc=   18 I=   18 A=   18 D=    1 M=    x
1765ns clock=1 pc=   19 I=60039 A=   18 D=    1 M=    x
1770ns clock=0 pc=   19 I=60039 A=   18 D=    1 M=    x
1775ns clock=1 pc=   18 I=   18 A=   18 D=    1 M=    x
1780ns clock=0 pc=   18 I=   18 A=   18 D=    1 M=    x
1785ns clock=1 pc=   19 I=60039 A=   18 D=    1 M=    x
1790ns clock=0 pc=   19 I=60039 A=   18 D=    1 M=    x
1795ns clock=1 pc=   18 I=   18 A=   18 D=    1 M=    x
1800ns clock=0 pc=   18 I=   18 A=   18 D=    1 M=    x
```

結語

您可以看到上述的輸出當中有 55 這個數字，這代表 1+...+10 = 55 的計算結果是正確的，也就

是處理器可以正常的執行 sum.hack 程式了。

雜誌訊息

讀者訂閱

程式人雜誌是一個結合「開放原始碼與公益捐款活動」的雜誌，簡稱「開放公益雜誌」。開放公益雜誌本著「讀書做善事、寫書做公益」的精神，我們非常歡迎程式人認養專欄、或者捐出您的網誌，如果您願意成為本雜誌的專欄作家，請加入 程式人雜誌社團 一同共襄盛舉。

我們透過發行這本雜誌，希望讓大家可以讀到想讀的書，學到想學的技術，同時也讓寫作的朋友的作品能產生良好價值 － 那就是讓讀者根據雜誌的價值捐款給慈善團體。 讀雜誌做公益也不需要有壓力，您不需要每讀一本就急著去捐款，您可以讀了十本再捐，或者使用固定的月捐款方式，當成是雜誌訂閱費，或者是季捐款、一年捐一次等都 OK！甚至是單純當個讀者我們也都很歡迎！

本雜誌每期參考價：NT 50 元，如果您喜歡本雜誌，請將書款捐贈公益團體。例如可捐贈給「羅慧夫顱顏基金會 彰化銀行(009) 帳號：5234-01-41778-800」。(若匯款要加註可用「程式人雜誌」五個字)

投稿須知

給專欄寫作者： 做公益不需要有壓力。如果您願意撰寫專欄，您可以輕鬆的寫，如果當月的稿件出不來，我們會安排其他稿件上場。

給網誌捐贈者： 如果您沒時間寫專欄或投稿，沒關係，只要將您的網誌以 [創作共用的「姓名標示、非商業性、相同方式分享」授權] 並通知我們，我們會自動從中選取需要的文章進行編輯，放入適當的雜誌當中出刊。

給文章投稿者： 程式人雜誌非常歡迎您加入作者的行列，如果您想撰寫任何文章或投稿，請用 markdown 或 LibreOffice 編輯好您的稿件，並於每個月 25 日前投稿到程式人雜誌社團 的檔案區，我們會盡可能將稿件編入隔月1號出版程式人雜誌當中，也歡迎您到社團中與我們一同討論。

如果您要投稿給程式人雜誌，我們最希望的格式是採用 markdown 的格式撰寫，然後將所有檔按壓縮為 zip 上傳到社團檔案區給我們， 如您想學習 markdown 的撰寫出版方式，可以參考 [看影片學 markdown 編輯出版流程] 一文。

如果您無法採用 markdown 的方式撰寫，也可以直接給我們您的稿件，像是 MS. Word 的 doc 檔或 LibreOffice 的 odt 檔都可以，我們 會將這些稿件改寫為 markdown 之後編入雜誌當中。

參與編輯

您也可以擔任程式人雜誌的編輯，甚至創造一個全新的公益雜誌，我們誠摯的邀請您加入「開放公益出版」的行列，如果您想擔任編輯或創造新雜誌，也歡迎到 程式人雜誌社團 來與我們討論相關事宜。

公益資訊

公益團體	聯絡資訊	服務對象	捐款帳號
財團法人羅慧夫顱顏基金會	http://www.nncf.org/ lynn@nncf.org 02-27190408分機232	顱顏患者 (如唇顎裂、小耳症或其他罕見顱顏缺陷）	銀行：009 彰化銀行民生分行 帳號：5234-01-41778-800
社團法人台灣省兒	http://www.cyga.org/ cyga99@gmail.com	單親、隔代教養.弱勢及一	銀行：新光銀行 戶名：台灣省兒童少年成長協會

童少年成長協會	04-23058005	般家庭之兒童青少年	帳號：103-0912-10-000212-0