การจัดองค์การคอมพิวเตอร์

Arithmetic Logic Unit

31110321 Computer Organization
สำหรับนักศึกษาชั้นปีที่ 3 สาขาวิชาวิศวกรรมคอมพิวเตอร์

ทรงฤทธิ์ กิติศรีวรพันธุ์

songrit@npu.ac.th

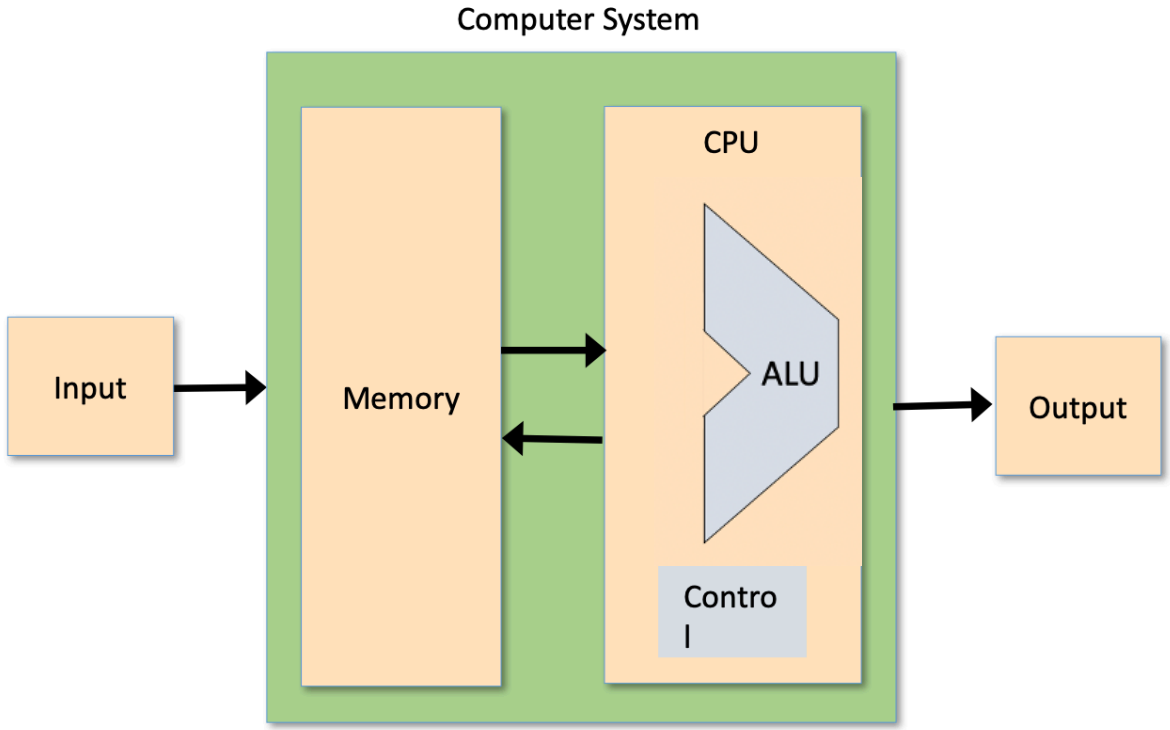
สาขาวิชาวิศวกรรมคอมพิวเตอร์
มหาวิทยาลัยนครพนม

Revised 2020-07-20

Outline

- 2.1 เลขไบนารี
- 2.2 การบวกเลขไบนารี
- 2.3 ตัวเลขลบ ในระบบดิจิทัล
- **2.4 Arithmetic Logic Unit (ALU)**
- 2.5 โปรเจ็คสัปดาห์ (พฤหัสบดี)
- 2.6 ภาพรวม

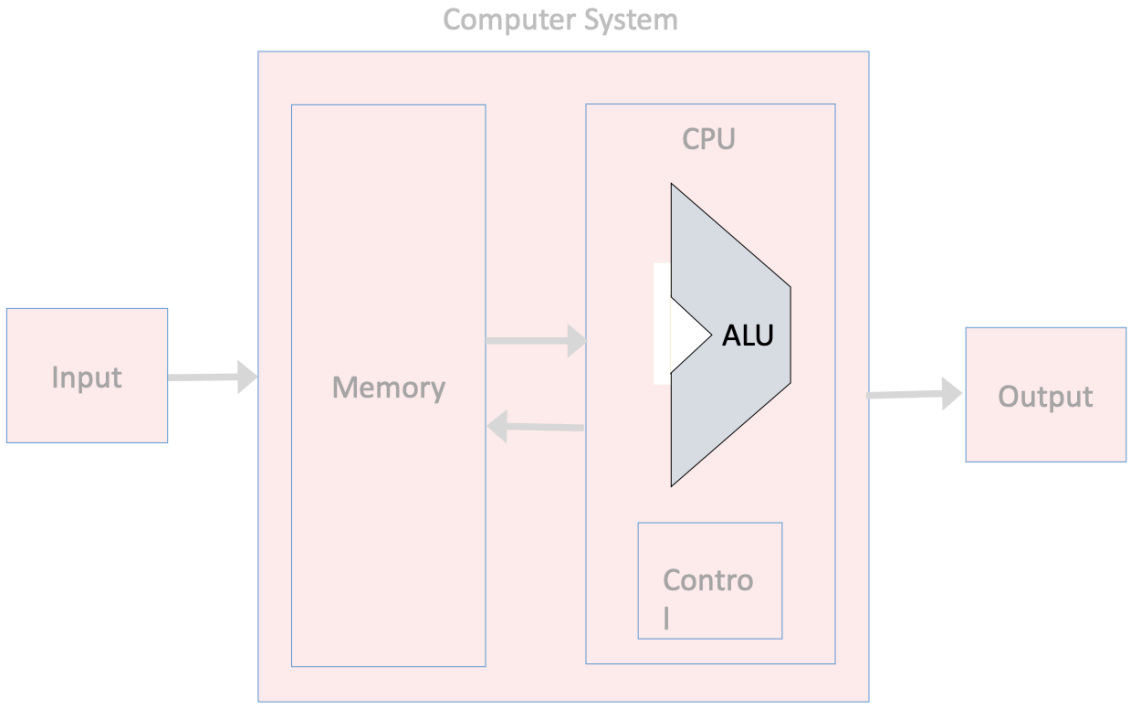
Von Neumann Architecture



- 1945 John Von Neumann
- เสนอสถาปัตยกรรมคอมพิวเตอร์สำหรับ
- ใช้งานได้หลากหลาย

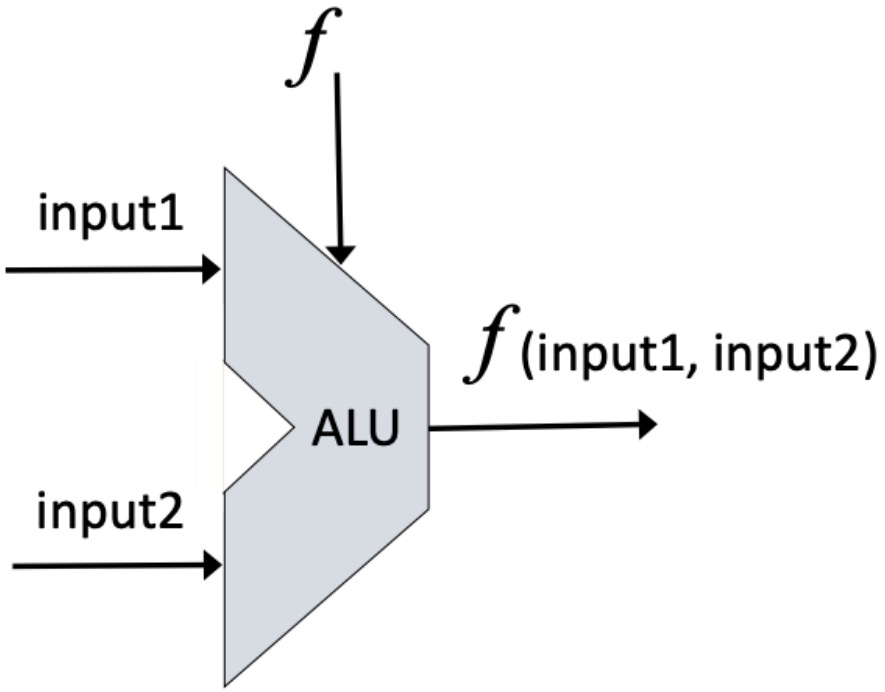
ส่วนประมวลผลคณิตศาสตร์

- อุปกรณ์สำคัญ ของสถาปัตยกรรมฟอนนอยด์มัน คือ
- **ส่วนประมวลผลคณิตศาสตร์**
- หรือ **เอเอลยู** (ALU)



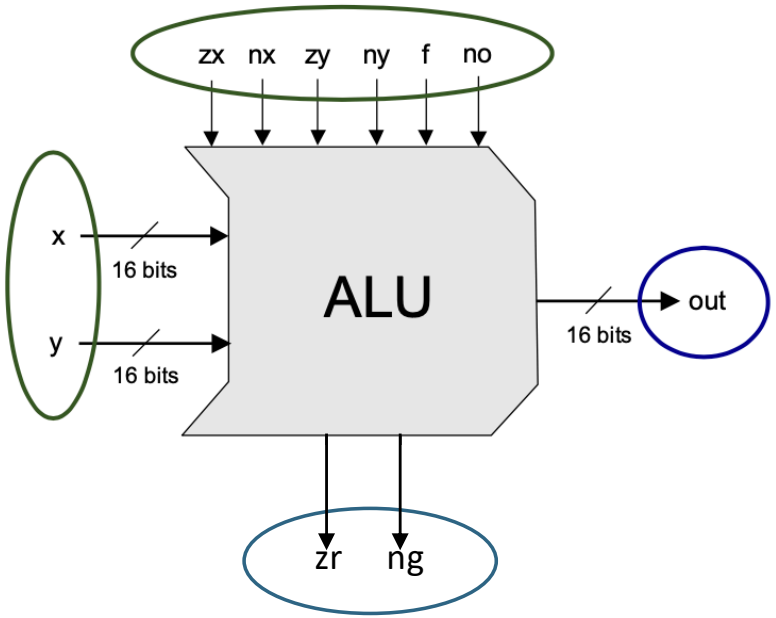
Arithmetic Logical Unit

- ส่วนคำนวณเลขไบนารี มีการรับข้อมูล 2 ชุด เพื่อนำข้อมูลนั้นมาประมวลผลกัน
- ป้อนคำสั่งประมวลที่ตำแหน่ง f
  - บวก
  - ลบ
  - เลื่อนบิต และ อื่น
- เมื่อคำนวนเสร็จส่งออก
  - F(input1, input2)



Hack ALU

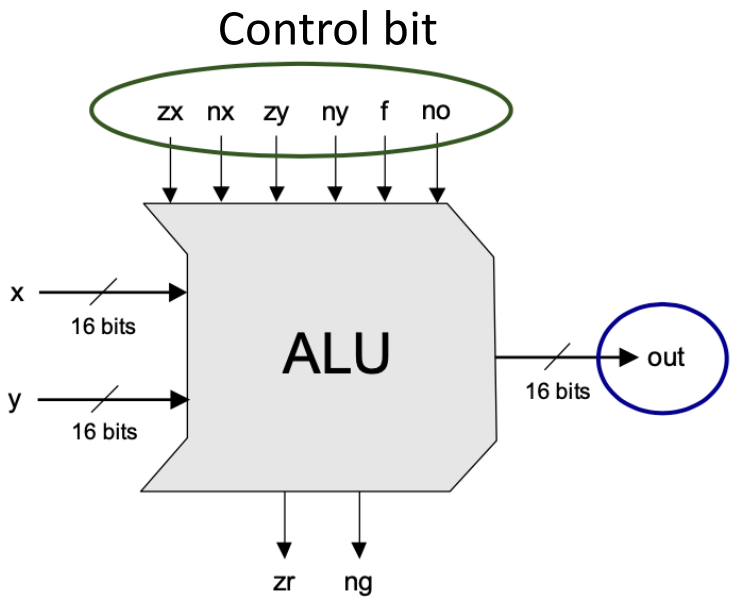
- คำนวณครั้งละ 16-bit
- คำนวณเลขลบ แบบ two's complement
- บัสอินพุท 16บิท 2 บัส
- มีขาส่ง**คำสั่ง** 6 บิท
- ส่งออกบัสเอาท์พุท 16บิท
- มี 2 บิทเอาท์พุท (zr, ng)



out
0
1
-1
x
y
!x
!y
-x
-y
x+1
y+1
x-1
y-1
x+y
x-y
y-x
x&y
x|y

Control Bit

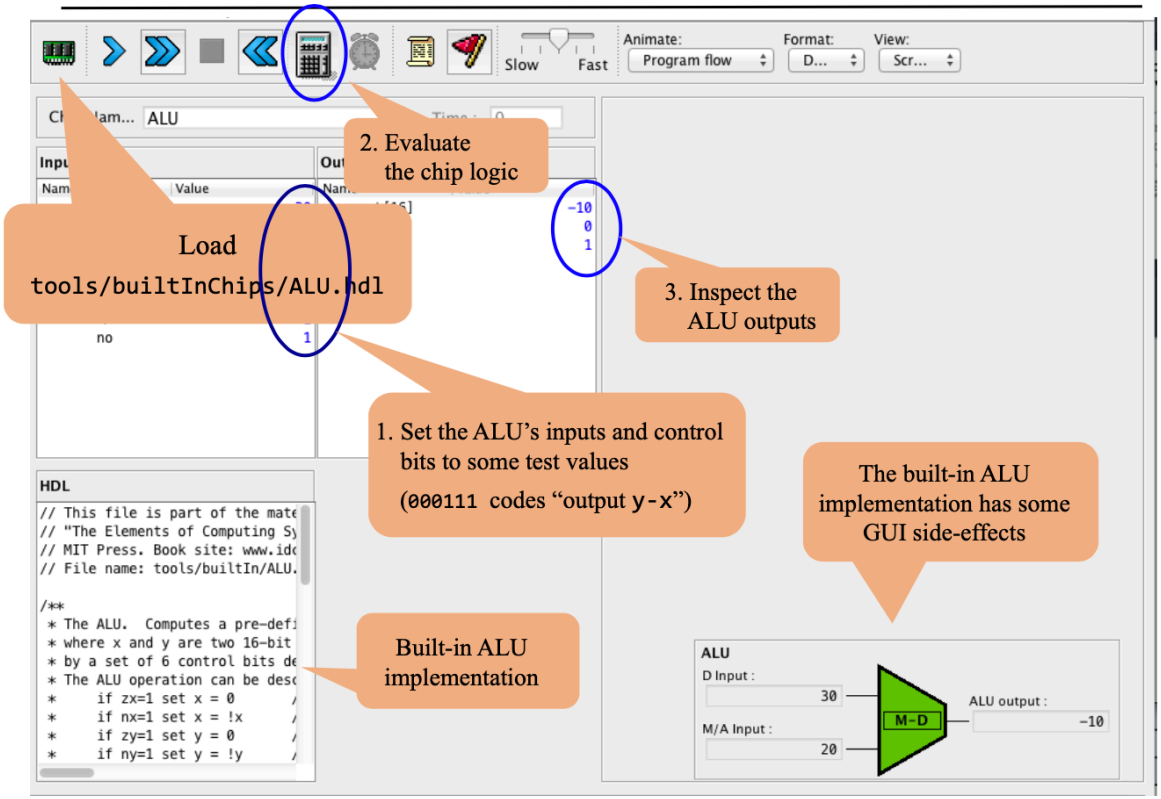
- ALU ทำงานด้วยการส่งคำสั่งใน Control Bit
- เราได้ออกแบบคำสั่ง Control bit เป็นตามตาราง



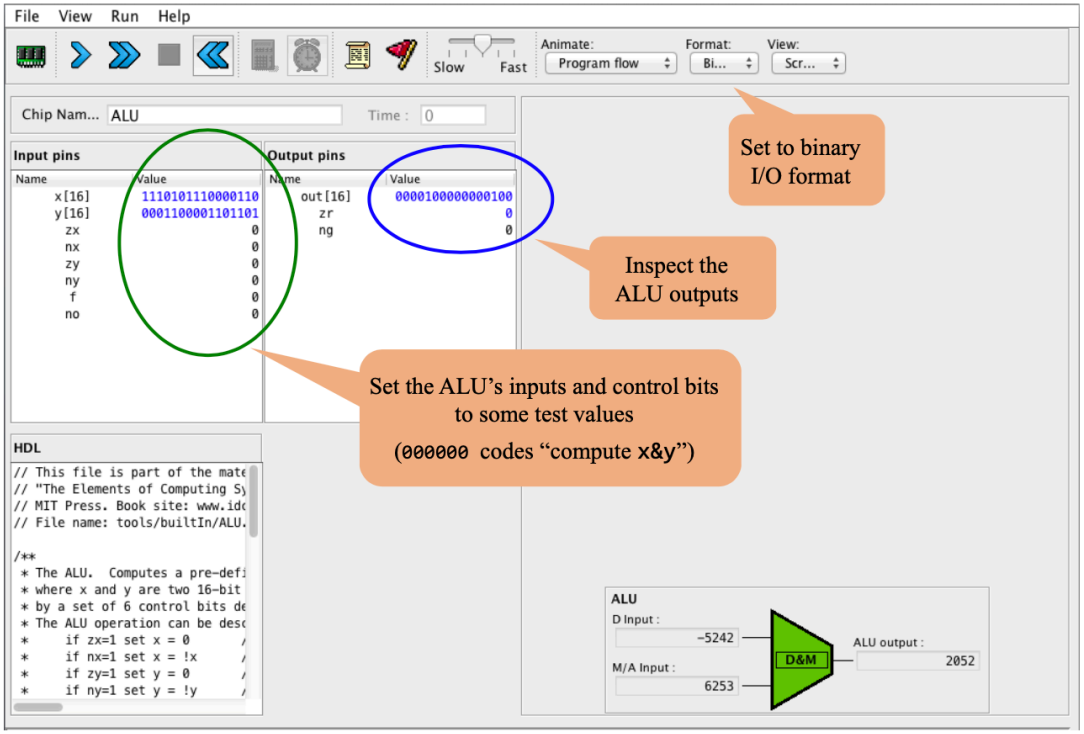
Control bit

zx	nx	zy	ny	f	no	out
1	0	1	0	1	0	0
1	1	1	1	1	1	1
1	1	1	0	1	0	-1
0	0	1	1	0	0	x
1	1	0	0	0	0	y
0	0	1	1	0	1	!x
1	1	0	0	0	1	!y
0	0	1	1	1	1	-x
1	1	0	0	1	1	-y
0	1	1	1	1	1	x+1
1	1	0	1	1	1	y+1
0	0	1	1	1	0	x-1
1	1	0	0	1	0	y-1
0	0	0	0	1	0	x+y
0	1	0	0	1	1	x-y
0	0	0	1	1	1	y-x
0	0	0	0	0	0	x&y
0	1	0	1	0	1	x|y

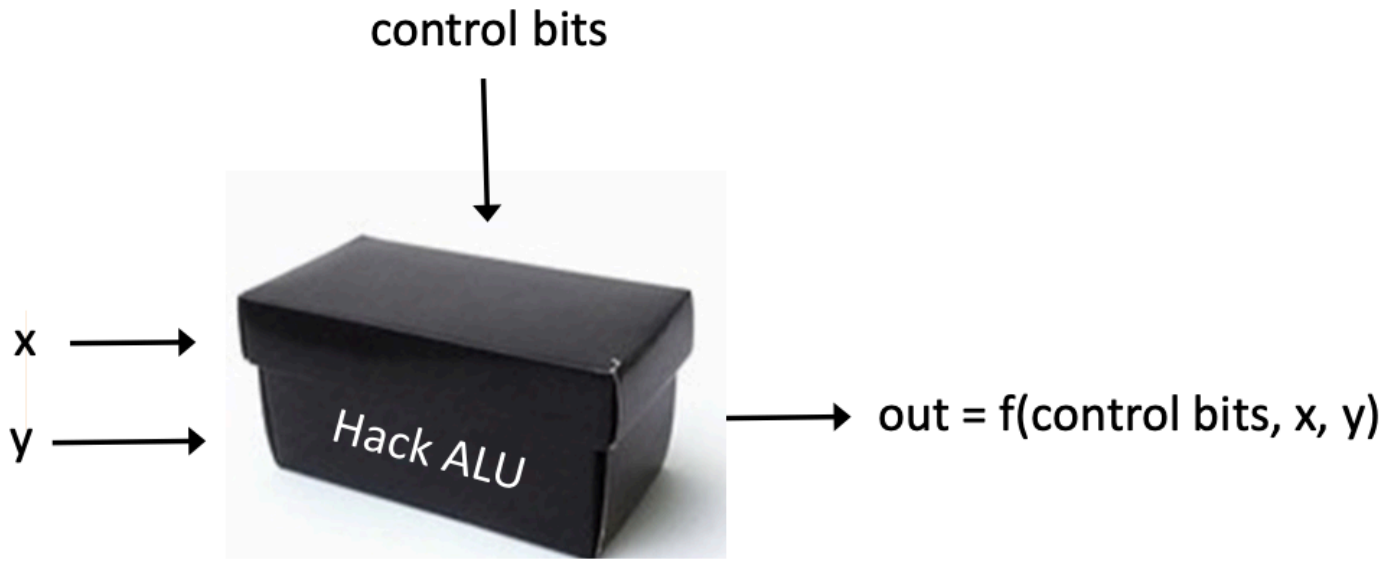
Hack ALU : คำนวณ y-x



Hack ALU : คำนวณ x&y

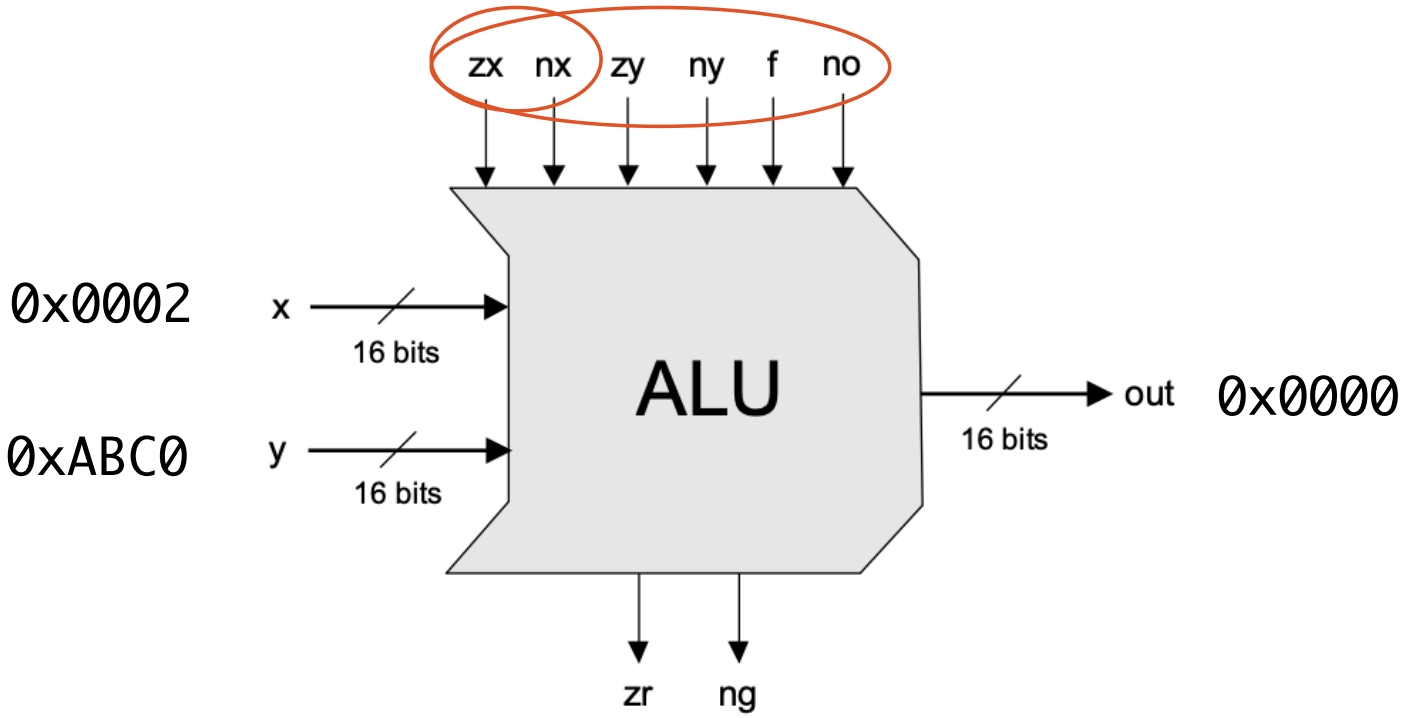


Hack ALU black box



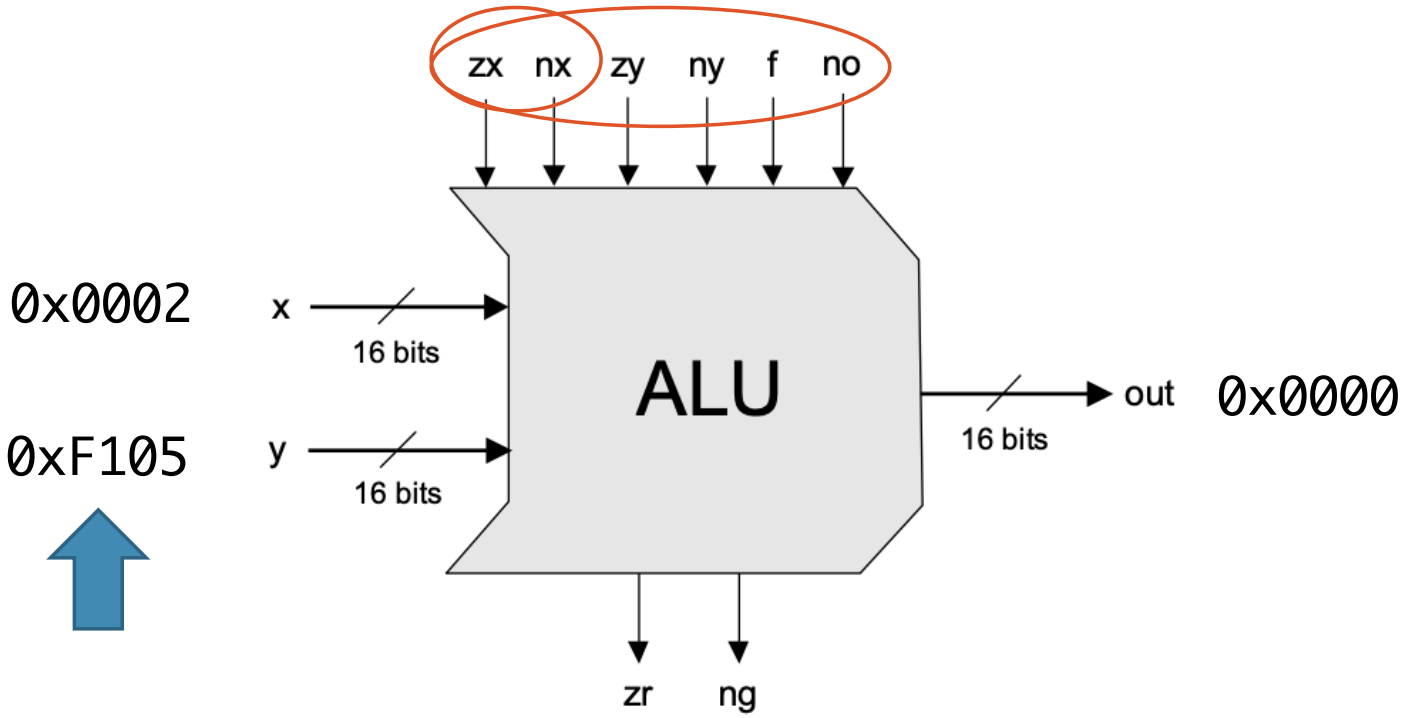
The Hack ALU operation

zx	nx	zy	ny	f	no	out
if zx then x=0	if nx then x=!x					



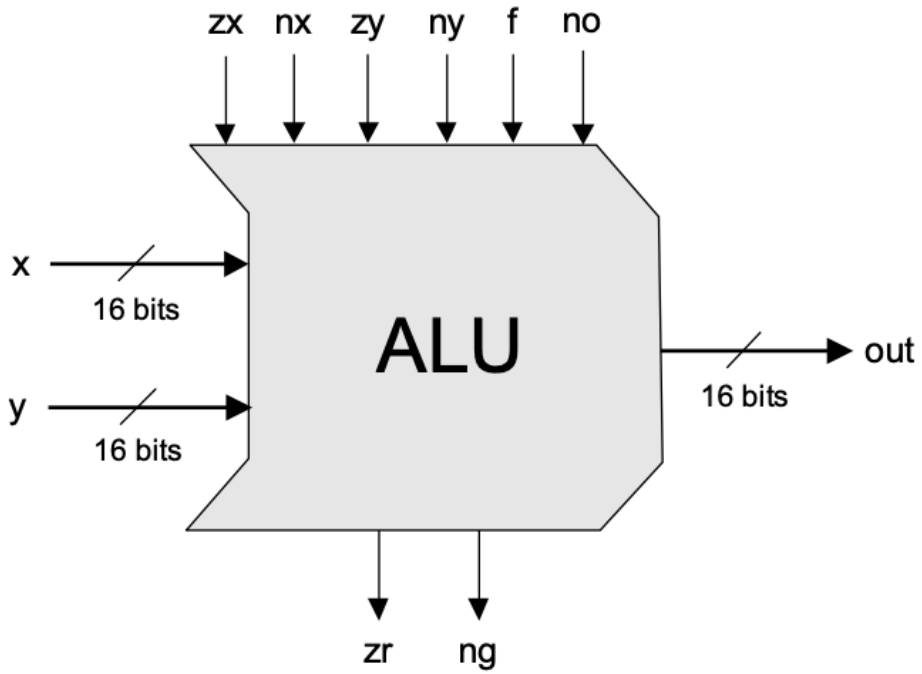
The Hack ALU operation

zx	nx	zy	ny	f	no	out
if zx then x=0	if nx then x=!x					



Hack ALU operation

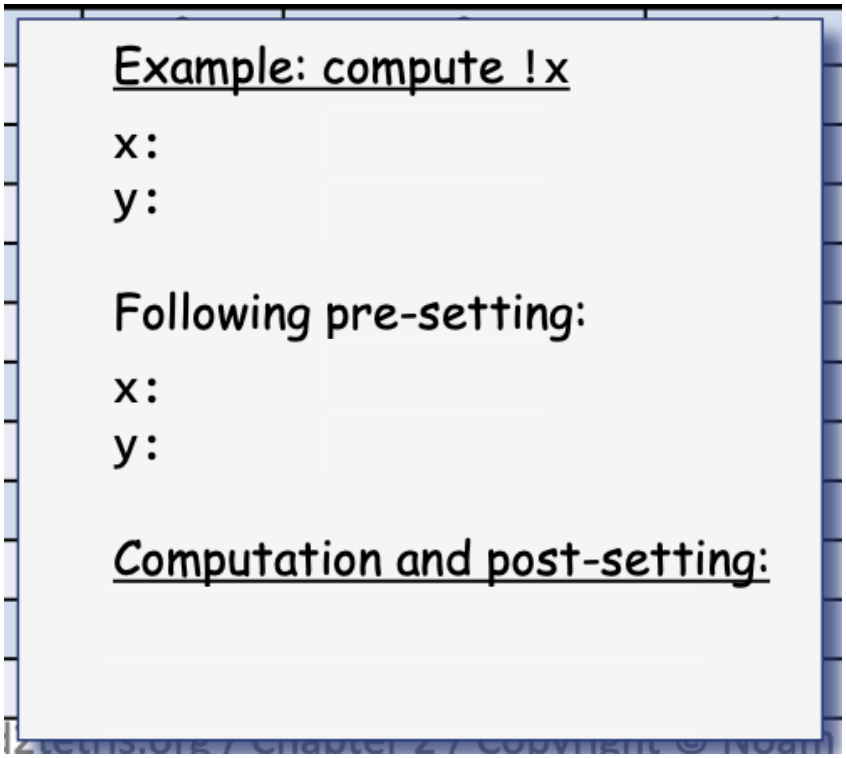
pre-setting the x input		pre-setting the y input		selecting between computing + or &	post-setting the output	Resulting ALU output
zx	nx	zy	ny	f	no	out
if zx then x=0	if nx then x=!x	if zy then y=0	if ny then y=!y	if f then out=x+y else out=x&y	if no then out=!out	out(x,y)=



Hack ALU operation

- !x
- inverse x

pre-setting the x input		pre-setting the y input		selecting between computing + or &	post-setting the output	Resulting ALU output
zx	nx	zy	ny	f	no	out
if zx then x=0	if nx then x=!x	if zy then y=0	if ny then y=!y	if f then out=x+y else out=x&y	if no then out=!out	out(x,y)=
1	0	1	0	1	0	0
1	1	1	1	1	1	1
1	1	1	0	1	0	-1
0	0	1	1	0	0	x
1	1	0	0	0	0	y
0	0	1	1	0	1	!x
1	1	0	0	0	1	!y
0	0	1	1	1	1	-x
1	1	0	0	1	1	-y
0	1	1	1	1	1	x+1
1	1	0	1	1	1	y+1
0	0	1	1	1	0	x-1
1	1	0	0	1	0	y-1
0	0	0	0	1	0	x+y
0	1	0	0	1	1	x-y
0	0	0	1	1	1	y-x
0	0	0	0	0	0	x&y
0	1	0	1	0	1	x|y



Hack ALU

Example: compute y-x

x:

y:

Following pre-setting:

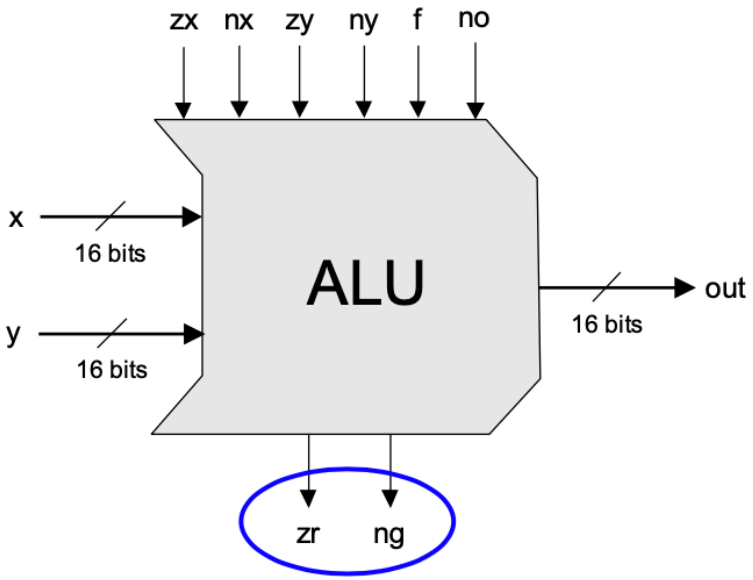
x:

y:

Computation and post-setting:

pre-setting the x input		pre-setting the y input		selecting between computing + or &	post-setting the output	Resulting ALU output
zx	nx	zy	ny	f	no	out
if zx then x=0	if nx then x=!x	if zy then y=0	if ny then y=!y	if f then out=x+y else out=x&y	if no then out=!out	out(x,y)=
1	0	1	0	1	0	0
1	1	1	1	1	1	1
1	1	1	0	1	0	-1
0	0	1	1	0	0	x
1	1	0	0	0	0	y
0	0	1	1	0	1	!x
1	1	0	0	0	1	!y
0	0	1	1	1	1	-x
1	1	0	0	1	1	-y
0	1	1	1	1	1	x+1
1	1	0	1	1	1	y+1
0	0	1	1	1	0	x-1
1	1	0	0	1	0	y-1
0	0	0	0	1	0	x+y
0	1	0	0	1	1	x-y
0	0	0	1	1	1	y-x
0	0	0	0	0	0	x&y
0	1	0	1	0	1	x|y

Hack ALU output control bits



- if (out == 0) then zr = 1, else zr = 0
- if (out < 0) then ng = 1, else ng = 0
- ได้ใช้ zr และ ng เมื่อขั้นตอนการสร้างร
  - แฮกค์คอมพิวเตอร์ ในบทที่ 4

Coming up: W2.5
Project

ภาพรวมโพรเจกต์ประจำสัปดาห์