كلية الهندسة
قسم هندسة الميكاترونيك

جَامعة المَنارة
MANARA UNIVERSITY

بنيان الحواسيب

محاضرة 5

Instructions: Language of the Computer

د. فادي متوج

مثال

- ما هي تعليمة MIPS التي يمثلها كود لغة الآلة التالي؟

op	rs	rt	rd	shamt	funct
0	8	9	10	0	34

1. `sub $t0, $t1, $t2`
2. `add $t2, $t0, $t1`
3. `sub $t2, $t1, $t0`
4. `sub $t2, $t0, $t1`

التعليمات المنطقية

على الرغم من أن أجهزة الكمبيوتر الأولى كانت تعمل على كلمات كاملة (مجموعة كاملة من 32 بت)، إلا أنه كان من المفيد العمل على حقول من البتات داخل الكلمة أو حتى على البتات الفردية. تسمى هذه التعليمات التي تتعامل مع البتات الفردية بالتعليمات المنطقية. يوضح الشكل التالي العمليات المنطقية في C و Java وما يقابلها في لغة التجميع MIPS

Logical operations	C operators	Java operators	MIPS instructions
Shift left	<<	<<	sll
Shift right	>>	>>>	srl
Bit-by-bit AND	&	&	and, andi
Bit-by-bit OR	|	|	or, ori
Bit-by-bit NOT	~	~	nor

التعليمات المنطقية
تعليمات الإزاحة

- الفئة الأولى من هذه العمليات المنطقية تسمى الإزاحة. تقوم هذه التعليمات بنقل كل البتات في كلمة ما إلى اليسار أو اليمين ،وملء البتات الفارغة بأصفار.
- يطلق على تعليمة الإزاحة المنطقية لليسار بـ (sll) shift left logical و يطلق على تعليمة الإزاحة المنطقية لليمين بـ shift right logical (srl)
- على سبيل المثال، إذا احتوى المسجل $s0 على القيمة :

0000 0000 0000 0000 0000 0000 0000 1001$_{two}$ = 9$_{ten}$

وتم تنفيذ تعليمة الإزاحة إلى اليسار بمقدار 4 ، ستكون القيمة الجديدة للمسجل $s0:

0000 0000 0000 0000 0000 0000 1001 0000$_{two}$= 144$_{ten}$

- تقوم التعليمة التالية بتنفيذ العملية السابقة ، بافتراض أن القيمة الأصلية كانت في السجل $s0 و أن النتيجة يجب أن يتم وضعها في المسجل $t2

```
sll $t2,$s0,4      # reg $t2 = reg $s0 << 4 bits
```

التعليمات المنطقية

تعليمات الإزاحة

```
sll $t2,$s0,4      # reg $t2 = reg $s0 << 4 bits
```

- هذه التعليمة تنتمي إلى النمط R-Type الذي مر معنا سابقاً و ترميز هذه التعليمة بلغة الآلة يعطى كما يلي:

op	rs	rt	rd	shamt	funct
0	0	16	10	4	0

- نلاحظ الحقل shmt الذي لم يستخدم في التعليمات الحسابية قد استخدم في تعليمة الإزاحة حيث تم وضع القيمة 4 فيه(مقدار الإزاحة)
- يوجد 0 في كل من حقلي op و funct، ويحتوي حقل rd على 10 ($t2)، ويحتوي rtعلى 16 ($s0) ، أما الحقل rs فهو غير مستخدم وبالتالي يتم وضعه على القيمة 0.
- توفر الإزاحة المنطقية إلى اليسار ميزة إضافية.حيث تعادل عملية الإزاحة إلى اليسار بمقدار i خانة عملية الضرب في 2^4
- على سبيل المثال تعليمة `sll $t2,$s0,4` الإزاحة المذكورة أعلاه تقوم بعملية إزاحة لليسار بمقدار 4 ، مما يعطي نفس نتيجة الضرب في 2^4 أو 16. حيث كانت القيمة قبل الإزاحة 9 ، وبعد الإزاحة أصبحت 9 × 16 = 144

التعليمات المنطقية

تعليمات AND ، OR ، NOR

هذه التعليمات تجري على مستوى الخانات الفردية (بت مقابل بت)

- مثال : بفرض المسجل $t2 يحوي القيمة 0000 0000 0000 0000 0000 1101 1100 0000two

و المسجل $t1 يحتوي على 0000 0000 0000 0000 0011 1100 0000 0000two

- يكون محتوى المسجل $t0 بعد تنفيذ التعليمة التالية

```
and $t0,$t1,$t2 # reg $t0 = reg $t1 & reg $t2
```

هو 0000 0000 0000 0000 0000 1100 0000 0000two

- يكون محتوى المسجل $t0 بعد تنفيذ التعليمة التالية

```
or $t0,$t1,$t2 # reg $t0 = reg $t1 & reg $t2
```

هو 0000 0000 0000 0000 0011 1101 1100 0000two

- يكون محتوى المسجل $t0 بعد تنفيذ التعليمة التالية

```
nor $t0,$t1,$t3 # reg $t0 = ~ (reg $t1 | reg $t3)
```

هو 1111 1111 1111 1111 1100 0011 1111 1111two

- توفر مجوعة تعليمات MIPS تعليمات andi و ori في حال احتوت العملية على قيمة ثابتة (قيمة فورية)

تعليمات اتخاذ القرارات
Instructions for Making Decisions

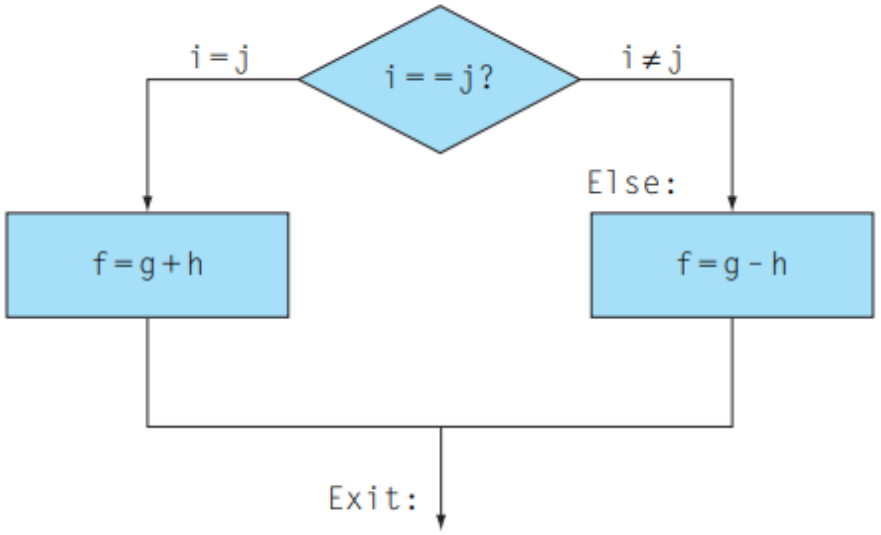
- ما يميز الكمبيوتر عن الآلة الحاسبة البسيطة هو قدرته على اتخاذ القرارات. بناءً على بيانات الإدخال والقيم التي تم إنشاؤها أثناء الحسابات ، يتم تنفيذ تعليمات مختلفة.
- يتم تمثيل عملية صنع القرار بشكل شائع في لغات البرمجة باستخدام عبارة if.
- تتضمن لغة التجميع MIPS تعليمين لاتخاذ القرار، يشبهان تعليمة if :
  - التعليمة الأولى هي: `beq register1, register2, L1`
    - ❖ تعني هذه التعليمة الانتقال إلى التعليمة المسماة L1 إذا كانت القيمة في `Register1` تساوي القيمة في `Register2`
    - ❖ كلمة `beq` هي اختصار للعبارة *branch if equal*
  - التعليمة الثانية هي : `bne register1, register2, L1`
    - تعني هذه التعليمة الانتقال إلى التعليمة المسماة L1 إ ذا كانت القيمة في `Register1` لا تساوي القيمة في `Register2`
    - كلمة `bne` هي اختصار للعبارة *branch if not equal*
- تسمى هاتان التعليماتان تعليمات التفرع المشروط أو تعليمات القفز المشروط (conditional branches)

تعليمات اتخاذ القرارات
Instructions for Making Decisions

مثال: ترجم تعليمة if-then-else التالية بلغة C إلى تعليمات قفز مشروطة بلغة التجميع

```
if (i == j) f = g + h; else f = g – h;
```

حيث المتغيرات الخمسة من f إلى j تتوافق مع المسجلات الخمسة $s0 إلى $s4



الحل:

```
      bne $s3,$s4,Else # go to Else if i ≠ j
      add $s0,$s1,$s2  #f=g+h(skipped if i≠j)
      j Exit           #go to Exit
Else: sub $s0,$s1,$s2  #f=g–h(skipped if i=j)
Exit:
```

الحلقات
Loops

نصادف الحلقات عند تكرار تنفيذ مجموعة من التعليمات أكثر من مرة مثل حلقة while في لغة C

مثال: ترجم حلقة while التالية المكتوبة بلغة C إلى لغة التجميع

```
while (save[i] == k)
        i += 1;
```

افترض أن i و k يقابلهما المسجلين $s3 و $s5 وأن العنوان الأساس للمصفوفة save موجودة في $s6

الحل:

```
Loop: sll $t1,$s3,2      # Temp reg $t1 = i * 4
      add $t1,$t1,$s6    # $t1 = address of save[i]
      lw $t0,0($t1)      # Temp reg $t0 = save[i]
      bne $t0,$s5, Exit  # go to Exit if save[i] ≠ k
      addi $s3,$s3,1     # i = i + 1
      j Loop             # go to Loop
Exit:
```

الحلقات
Loops

- ربما يكون اختبار المساواة أو عدم المساواة هو الاختبار الأكثر شيوعًا، ولكن في بعض الأحيان يكون من المفيد معرفة ما إذا كان متغير ما أقل من متغير آخر.
- ويتم إجراء مثل هذه المقارنات بلغة التجميع MIPS باستخدام تعليمة تقارن بين مسجلين و وضع القيمة 1 في مسجل ثالث إذا كانت قيمة المسجل الأول أقل من قيمة المسجل الثاني؛ وإلا، يتم وضع القيمة 0 فيه.
- تدعى تعليمة MIPS التي تقوم بذلك `slt` و تعني set on less than
- على سبيل المثال `slt $t0,$s3,$s4 #$t0 = 1 if $s3 < $s4`

يعني أن المسجل $t0 يتم ضبطه على القيمة 1 إذا كانت القيمة في المسجل $s3 أقل من القيمة في المسجل $s4 و ماعدا ذلك، المسجل $t0 يتم ضبطه على القيمة 0.

- هناك نسخة فورية من تعليمة `slt`. مثلاً لاختبار ما إذا كانت قيمة مسجل `$s2` أقل من الثابت 10 يتم ذلك من خلال كتابة التعليمة التالية:

```
slti $t0,$s2,10 # $t0 = 1 if $s2 < 10
```

مقارنة الأعداد بإشارة و بدون إشارة

- لمقارنة الأعداد بإشارة : set on less than immediate (slti) ، Set on less than (slt)
- لمقارنة الأعداد بدون إشارة : set on less than immediate unsigned (sltiu) ، set on less than unsigned(sltu)

مثال :

لنفترض أن المسجل $s0 يحتوي على الرقم الثنائي

1111 1111 1111 1111 1111 1111 1111 1111two

وأن المسجل $s1 يحتوي على الرقم الثنائي

0000 0000 0000 0000 0000 0000 0000 0001two

ما هي قيم المسجلات $t0 و $t1 بعد تنفيذ التعليمتين التاليتين؟

```
slt $t0, $s0, $s1 # signed comparison
sltu $t1, $s0, $s1 # unsigned comparison
```