

الفريق العربي للهندسة العكسية

دورة ATRE لتعليم الأسس للمبتدئين من الصفر الدرس الأول – أنظمة العد Number Systems

سنتكلم اليوم عن أنظمة العد الأربعة الأساسية ...

: Binary

هذا هو النظام الثنائي وهو اللغة الأم للحاسوب... فالحاسوب من الداخل مكون من دوائر كهربائية... هذه الدوائر لا تعرف الـ assembly أو الـ visual basic أو أي شيء من هذه اللغات... هي تتعامل مع التيار الكهربائي... وجود تيار كهربائي أو عدم وجود كهربائي ، وقد اتفق على أن يرمز لحالة وجود التيار بالرمز 1 وعدم وجوده بالرمز 0 .
النظام الثنائي إذن يتكون من رقمين فقط هما 0 و 1 ... وبالتالي فإن أساس النظام (Base). يساوي 2 .
مثلا فإن تعليمة مثل هذه :

```
MOV EAX,DWORD PTR DS:[40A710h]
```

هي بالأصل يتم التعامل معها بداخل الدوائر الكهربائية للحاسوب على أنها

```
1010000100010000101001110100000000000000
```

أما تعليمة

```
Push EAX
```

فهي

```
1010000
```

والآن دعنا نرى... كيف يتم العد بالنظام الثنائي؟ دعني أذكرك... بالنظام العشري... كنا نعد هكذا: 1 2 3 ... 9 10 ، هل لاحظت ؟
النظام العشري مكون من 10 رموز هي 0123456789 ، إذن بعد الوصول إلى 9 فإن الرموز (الأرقام) تنتهي... ما العمل؟ نبدأ من جديد فتصبح الـ 9 صفرا ، ونضيف 1 إلى الخانة التالية (خانة العشرات) ، إذن 9 يليها 10... بالمثل عندما نصل إلى 399 فإن خانة الأحاد قد وصلت إلى آخر رقم وهو 9 إذن نعيد العد ونستبدل الـ 9 بصفر ونضيف واحد للخانة التالية... الخانة التالية بها 9 ان أضفت إليها 1 ستصبح 10 إذن سنكتب اصفر والواحد ينتقل للخانة التالية وهي الـ 3 ، فتصبح 4 ، أي سنصل إلى 400 .

تابع معي كيف نعد بالنظام الثنائي :

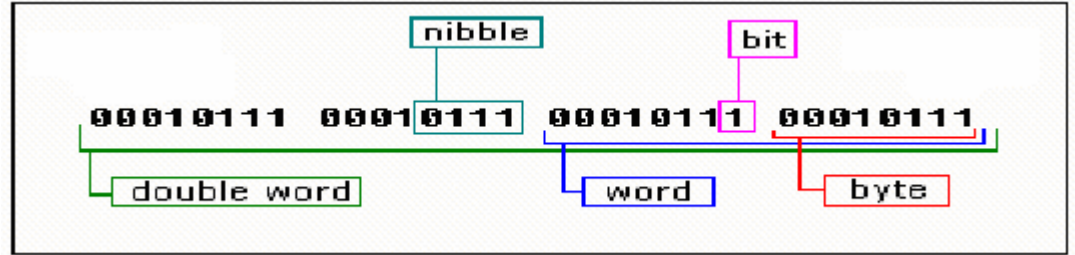
0 ثم 1 ثم 10 . ما الذي حصل؟ كنا نعد بشكل طبيعي... 0... يليه 1... يليه ماذا ؟ لا يليه شيء فالواحد هو آخر رقم في هذا النظام. إذن استبدله بصفر وأضف واحد للخانة الجديدة فتحصل على 10 . فلنتابع : 10 ثم 11 ثم 100 ، ماذا حصل ؟ الـ 10 يليها 11.. يليها 100 . كما ترى فخانة الأحاد قد وصلت إلى آخر عدد إذن استبدله بصفر وأضف واحد للخانة التالية (أي خانة العشرات) ، لكن خانة العشرات بدورها قد وصلت لآخر عدد.. إذن عندما نضيف 1 للـ 1 سنحصل على 10 أي نبقى الصفر وننقل الـ 1 للخانة التالية فيصبح لدينا 100.

انظر :

decimal	binary
0	0
1	1
2	10
3	11
4	100
5	101
6	110
7	111
8	1000
9	1001

أرجو أن تكون الفكرة قد أصبحت واضحة.

كل رقم (digit) في النظام الثنائي يسمى **BIT** ، كل 4 bits تسمى **NIBBLE** ، كل 8 bits تكون **BYTE** ، كل two bytes تكون **WORD** ، وختاما كل two words تكون **DOUBLE WORD** .
انظر الشكل التالي :



إن أول بت (في اليمين) يسمى low bit أو Least Significant Bit (LSB) أي البت ذو القيمة الأقل ، وفي المقابل فإن آخر بت (في اليسار) يسمى high bit أو Most Significant Bit (MSB) أي البت ذو القيمة الأعلى.

ملاحظة : عندما نقول $10100101b$ فإننا نعني قيمة بالباينري ولهذا وضعنا حرف b في نهاية الرقم... لكن هذا ليس شرطا...

ماذا إذا أردنا جمع عددين بالنظام الثنائي ؟ لا بأس دعنا نرى...

$$0=0+0$$

$$1=1+0$$

$$10=1+1$$

والآن لنرى كيف نجمع 1100110 مع 1101

$$\begin{array}{r} \textcircled{1} \quad \textcircled{1} \\ 1 \ 1 \ 0 \ 0 \ 1 \ 1 \ 0 \\ 0 \ 0 \ 0 \ 1 \ 1 \ 0 \ 1 \quad + \\ \hline 1 \ 1 \ 1 \ 0 \ 0 \ 1 \ 1 \end{array}$$

كما ترى ، فقد أضفنا أصفارا للرقم السفلي... هذا للتوضيح فقط... كي تسهل علينا عملية الحساب... لأنك تعلم ان الصفر على اليسار لا قيمة له...

أول صفر مع أول 1 ننتجتهم 1

ثاني 1 مع الـ 0 ننتجتهم 1

ثالث 1 مع الـ 1 ننتجتهم 10 - إذن نكتب الـ 0 ونضيف 1 للخانه التالية

رابع 0 مع الـ 1 مع الـ 1 السابق ننتجتهم 10 - إذن نكتب الـ 0 ونضيف 1 للخانه التالية

خامس 0 مع الـ 0 مع الـ 1 السابق ننتجتهم 1

سادس 1 مع الـ 0 ننتجتهم 1

سابع 1 مع الـ 0 ننتجتهم 1 .

لن أشرح العمليات الأخرى أي الطرح والضرب والقسمة فهذه ستأخذ وقتا طويلا إضافة إلى انه لا فائدة كبيرة منها...

النظام السداسي العشري Hexadecimal

يتكون هذا النظام من 16 رقم (رمز) هم : (من اليمين لليساار)

F E D C B A 9 8 7 6 5 4 3 2 1 0

أساس النظام (Base) يساوي 16. دعنا نرى كيف يمكن أن نعد باستخدام هذا النظام...

0 ثم 1 ثم 2 ... ثم 9 ثم A ثم B ... ثم F ثم 10

ما الذي حصل؟ كما في السابق..بدأنا العد وحين انتهت الأرقام ووصلنا إلى آخر رقم ممكن وهو F اضطررنا إلى استبداله بصفر وإضافة 1 للخانة التالية.

انظر إلى هذا : (ابدأ التتبع من اليسار)

... 108 109 10A 10B 10C 10D 10E 10F 110 111 112 113....

... D389 D38A D38B D38C D38D D38E D38F D390...

كما في النظام الثنائي فلن أتطرق لعمليات الحساب في هذا النظام رغم أنها تتبع نفس القاعدة في النظام السابق....

النظام الثماني OCTAL

هذا النظام يتكون من ثمانية رموز هي 0 1 2 3 4 5 6 7

والعد فيه يكون كالتالي (من اليسار لليمين)

0 1 2 3 4 5 6 7 10 11 12 13 14 15 16 17 20 21 22 23 24 25 26 27 30 31 ...
32 33 ...

النظام العشري Decimal

هذا هو النظام المألوف لجميع الناس...في النظام العشري هناك 10 أرقام (digits) :

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

التحويل من أي نظام إلى Decimal

للتحويل من أي نظام إلى النظام العشري فإننا نبدأ بالـ MSB ونضربه بأساس النظام مرفوعا إلى القوة التي تساوي ترتيب الرقم ، ثم ننتقل إلى الرقم التالي فنضربه بالأساس مرفوعا إلى القوة التي تساوي ترتيبه...ونجمع هذه القيم.

مثلا لتحويل 10100101b من النظام الثنائي إلى العشري ،

$$\begin{aligned} 10100101b &= \\ &= 1 \cdot 2^7 + 0 \cdot 2^6 + 1 \cdot 2^5 + 0 \cdot 2^4 + 0 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 \\ &= 128 + 0 + 32 + 0 + 0 + 4 + 0 + 1 = 165 \\ &\quad \text{(decimal value)} \end{aligned}$$

base

digit position

كما ترى...للتحويل من النظام الثنائي إلى العشري فإننا نبدأ بالـ MSB ونضربه ب 2 (الأساس) مرفوعة للقوة التي تساوي ترتيبه ، في حالتنا 7 ، ثم نجمع عليها البت التالي مضروبا ب 2 مرفوعة للقوة التي تساوي ترتيبه...وهكذا...تمعن في الصورة فهي واضحة...

دعنا نطبق الفكرة على تحويل القيمة 1234h من سداسي عشري إلى عشري :

$$1 \cdot 16^3 + 2 \cdot 16^2 + 3 \cdot 16^1 + 4 \cdot 16^0 = 4096 + 512 + 48 + 4 = 4660$$

(decimal value)

base
digit position

نبدأ بالـ MSB وهو 1 ، نضربه بالأساس (أي 16) مرفوعاً إلى ترتيب الرقم (أي 3) ، ثم نكمل فنجمعه مع 2 مضروبة بالأساس (أي 16) مرفوعاً إلى الترتيب (أي 2) وهكذا...
نفس الفكرة تنطبق على التحويل من النظام الثماني إلى النظام العشري.

التحويل من Decimal إلى أي نظام آخر :

الفكرة هي : نقوم بقسمة القيمة العشرية على أساس النظام الذي نريد التحويل عليه ، ولا تنسى أن هناك باقٍ . تابع عملية القسمة إلى أن يصبح ناتج القسمة صفراً...
سنطبق الفكرة على التحويل من Decimal إلى Binary :
دعنا نقوم بتحويل القيمة 359 إلى ثنائي :

Divider	reminder	notes
359/2=179	1	بدئنا بقسمة الـ 359 على 2 والناتج 179.5 أي 179 والباقي 1
179/2=89	1	والآن 179 تقسيم 2 يساوي 89.5 أي 89 والباقي 1
89/2=44	1	89 تقسيم 2 يساوي 44.5 أي 44 والباقي 1
44/2=22	0	44 تقسيم 2 يساوي 22 والباقي 0
22/2=11	0	22 تقسيم 2 يساوي 11 والباقي 0
11/2=5	1	11 تقسيم 2 يساوي 5.5 أي 5 والباقي 1
5/2=2	1	5 تقسيم 2 يساوي 2.5 أي 2 والباقي 1
2/2=1	0	2 تقسيم 2 يساوي 1 والباقي 0
1/2=0	1	1 تقسيم 2 يساوي 0 والباقي 1 ... نتيجة القسمة 0 إذن فقد انتهينا

والآن الجواب هو ما تراه في عمود reminder من أسفل لأعلى ، من اليسار لليمين . أي أن

$$359=101100111$$

دعنا نطبق الفكرة على التحويل من النظام السداسي العشري إلى النظام العشري.
نريد تحويل 67024 إلى النظام العشري.

Divider	reminder	notes
67024/16=4189	0	67024 على 16 يساوي 4189 والباقي 0
4189/16=261.8125	0.8125 x 16 = 13 → D	4189 تقسيم 16 يساوي 261 والباقي 0.8125 ، نقوم بضرب الباقي بـ 16 فنحصل على 13 ، أي D
261/16=16.3125	0.3125 x 16 = 5	261 تقسيم 16 يساوي 16 والباقي 0.3125 ، نقوم بضرب الباقي بـ 16 فنحصل على 5
16/16=1	0	16 تقسيم 16 يساوي 1 والباقي 0
1/16=0	1	1 تقسيم 16 يساوي 0 والباقي 1 ، الناتج 0 إذن انتهينا.

والآن الجواب هو ما تراه في عمود reminder من أسفل لأعلى ، من اليسار لليمين . أي أن

$$67024 = 105D0$$

ملاحظة : عند البرمجة بالاسمبلي فيجب أن يسبق أي قيمة بالسداسي عشري ، الرقم 0 إذا كانت مبتدئة بحرف... فمثلا القيمة 1B2 تبقى كما هي لأنها ليست مبتدئة بحرف ، لكن C441 يجب أن تكتبها 0C441 .

التحويل من و إلى النظام Binary والـ Hex .

الأمر سهل للغاية.. أنظر الجدول بالأسفل... لكل رقم بالهكس هناك ما يقابله بالنظام الثنائي (مع الوقت يمكنك حفظه لكن هذا ليس مطلوباً منك) . مثلاً افترض أننا نريد التحويل من 1001101101b إلى هكس... عليك أن تبدأ من LSB ، قم بتقسيم الرقم إلى مجموعات كل مجموعة مكونة من 4 خانات. هكذا :

(لاحظ أن المجموعة الثالثة في اليسار مكونة من رقمين فقط لكن نحن نريدها مكونة من 4 أرقام ، لذا نضيف إليها صفرين إلى اليسار)

0 0 1 0 0 1 1 0 1 1 0 1

والآن من الجدول يمكنك أن تلاحظ أن 1101 تقابل D ، و 0110 تقابل 6 ، أما 0010 فتقابل 2
أذن الناتج 26D .

والعكس يتم بنفس الطريقة ، لتحويل 0B45A نستبدل كل رقم بما يقابله ، ف A يقابلها 1010 و 5 يقابلها 0101 و 4 يقابلها 0100 أما B فيقابلها 1011 ، إذن النتيجة 1011010001011010

Binary	Decimal	Octal	Hex
0000	0	0	0
0001	1	1	1
0010	2	2	2
0011	3	3	3
0100	4	4	4
0101	5	5	5
0110	6	6	6
0111	7	7	7
1000	8	10	8
1001	9	12	9
1010	10	13	A
1011	11	14	B
1100	12	15	C
1101	13	16	D
1110	14	17	E
1111	15	21	F

Signed Numbers :

تمعن في هذه القيمة : 0B521... هل هي موجبة أم سالبة ؟ بالطبع لا مجال للحديث عن الموجب أو السالب فهذه قيمة لا تحمل إشارة...

المكمل الأول: First complement

إذا كان لدينا 8-bits (بايت) فيمكننا عمل 256 عدد مختلف (256 combinations) بما فيها الصفر. إذن يمكننا أن نفترض أن أول 128 عدد (0-127) هي موجبة والأعداد التالية (128-256) سالبة. للحصول على ال-1st complement لأي عدد موجب فإننا ببساطة نحوله إلى ما يقابله بال- binary . إذن للحصول على ال-1st c للقيمة 54 فإننا نحولها إلى binary فنحصل على 110110

أما لذا كانت القيمة سالبة ، أي 54- فإننا نطبق المعادلة التالية :

$$1^{\text{st}} c = 2^{\text{power } n} - 1 - \text{number in binary}$$

n التي تراها تعني عدد الخانات ، فإن كانت القيمة بايت فـ n=8 وأن كانت word فـ n=16 وهكذا...

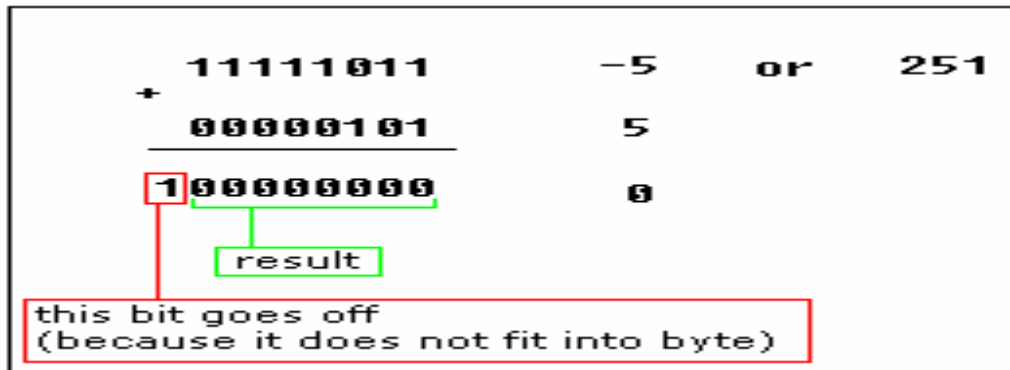
2 power n أي 2 مرفوعة للأس n .

Number in binary أي الرقم بعد تحويلها إلى binary . لنطبق المعادلة على القيمة 54- لتحويلها إلى 1st c بصيغة بايت

$$1^{\text{st}} c = 2^{\text{power } 8} - 1 - 110110 \\ = 11111111 - 110110 = 11001001$$

هناك طريقة أخرى أسهل من هذه... لو لاحظت فإن القيمة 54 بالتنائي تساوي 00110110 (أضفنا صفرين لليسار كي يصبح العدد مكونا من 8 خانات) والآن اعكس كل بت فتحصل على 11001001 . هل رأيت ما أسهلها :

لكن دعنا نتأمل قليلا... لماذا تم اختيار هذه الطريقة (أقصد طريقة ال-1st complement) وليس أي طريقة أخرى ؟ إذا جمعت 5- مع 5 فستحصل على 0 . أليس كذلك؟ حسنا هذا يحدث عندما يجمع ال-processor القيمة 5 مع القيمة 251 ، النتيجة هي 255 ، وبسبب ال-overflow فالنتيجة تعتبر صفرا .



كما ترى الناتج هو 100000000 لكن نحن جمعنا عددين من فئة byte أي 8bit لذا يجب أن يكون الجواب مكون أيضا من 8bit لكن حيث أننا حصلنا على 9bit فإن هذه الحالة تسمى في علم الحاسوب overflow ، ويتم تجاهل ال-MSB ويبقى 00000000 أي صفر...

نفس المبدأ ينطبق عند التعامل ما فئة word (16 bit values) حيث أن 16 bits تنشئ 65536 عددا مختلفا ، أول 32768 عدد (من 0 إلى 32767) تستخدم لتمثيل الأعداد الموجبة ، والباقي (32768 إلى 65536) تستخدم لتمثيل الأعداد السالبة.

المكمل الثاني: Second complement

المكمل الثاني للأعداد الموجبة هو نفسه المكمل الأول والذي بدوره عبارة عن العدد بعد تحويله للـ binary .
فأما 54 عند تحويلها للمكمل الثاني نحصل على 110110
أما المكمل الثاني للقيم السالبة فيتم الحصول عليه بإضافة 1 إلى المكمل الأول.
في المثال السابق عرفنا أن 54- بالمكمل الأول تساوي 11001001 إذن بالمكمل الثاني :

$$2^{\text{nd}} c = 1^{\text{st}} c + 1 = 11001001 + 1 = 11001010$$

لن أتعمق أكثر من هذا...يمكنني كتابة عشرات الصفحات كي أشرح هذا النظام لكن هذا ليس محور حديثنا.

Assignment :

I-convert these values from decimal to hexadecimal , then to Binary

- a) 15 b)150

II-convert these values from Binary to Decimal , then to Hex.

- a) 1001 b)011101101

III-convert these values from Hex to Decimal , then to Binary.

- a) 0B51 b)124C

IV- find the 1st and 2nd complement of the following decimal values :

- a) 154
b) - 413

solutions should be mailed to revenge-34[at]hotmail[dot]com .

deadline: one week from the lesson's date .

References:

- 1- http://en.wikipedia.org/wiki/Main_Page
- 2- <http://www.al-ebda3.info/ib/>
- 3- <http://www.emu8086.com/>
- 4- <http://www.arabteam2000.com/>
- 5- Assembly Language: Step-by-Step , Jeff Duntemann
- 6- introduction to computer , Jawdat Abu Taha

Arab Team for Reverse Engineering

αλλκο
July - 2006