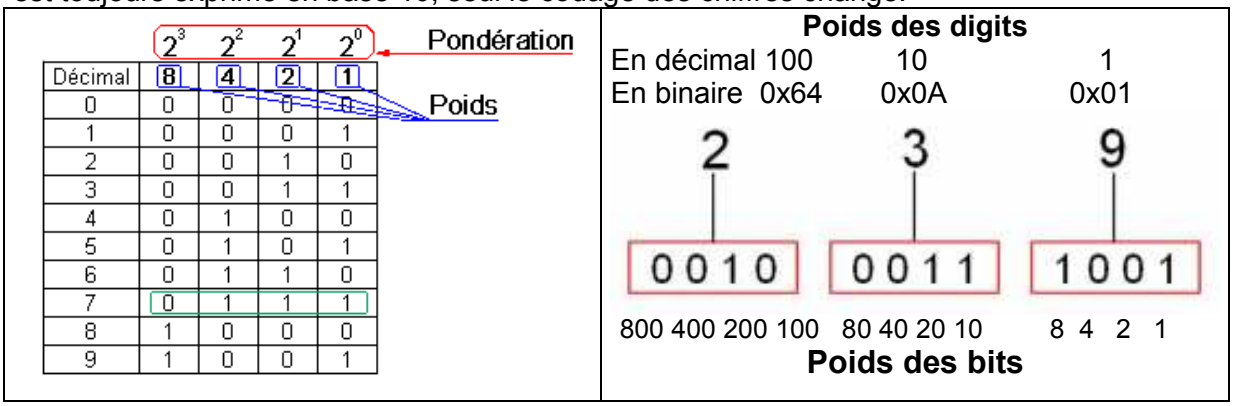
CH-1092 Belmont, Switzerland
Tel +41 21 728-6156, Fax 728-6157
Email info@didel.com, www.didel.com

BCD – Décimal codé binaire

Le code BCD fait correspondre aux chiffres de 0 à 9 leur équivalent binaire 4 bit. Le nombre est toujours exprimé en base 10, seul le codage des chiffres change.



Décimal	2^3 8	2^2 4	2^1 2	2^0 1
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

Les opérations sur des nombres sont efficaces en binaire. Si les données sont en BCD et les résultats aussi en BCD, c'est un peu lourd de faire une double conversion, mais c'est ce qui se fait en général. Les opérations directes sur les nombres BCD sont naturellement plus efficaces.

Incrémenter

Pour compter en BCD, on compte en binaire, et on détecte la combinaison 1010 qui suit 1001=9 et on ajoute 6, ce qui corrige le digit et ajoute 1 au chiffre de poids plus fort.

```
// IncBCD.ino   fonction et test
void setup()  {
  Serial.begin (9600);
}
unsigned int IncBCD ()  {  // ajoute 1
unsigned int bcd;
    if ((bcd & 0x000F) > 0x9) { bcd += 0x6; }
    if ((bcd & 0x00F0) > 0x90) { bcd += 0x60; }
    if ((bcd & 0x0F00) > 0x900) { bcd += 0x600; }
    if ((bcd & 0xF000) > 0x9000) { bcd += 0x6000; } // déborde!
    return bcd;
 }
unsigned int var;
void loop()  {    // test avec 2 valeurs
  var = IncBCD(0x299) ;
  Serial.print (var,HEX);
   var = IncBCD(0x9999) ;
  Serial.println (var,HEX);
  delay (2000);
}
```

Décrémenter

Pour décompter en BCD, il suffit de surveiller les digits valant 0xF résultants de la soustraction.

```
// DecBCD.ino   fonction et test
. . .
unsigned int DecBCD (unsigned int bcd)  {  // ajoute 1
unsigned int outBCD;
    bcd--;
    if ((bcd & 0x000F) == 0xF) { bcd -= 0x; }
    if ((bcd & 0x00F0) == 0xF0) { bcd -= 0x60; }
    if ((bcd & 0x0F00) == 0xF00) { bcd -= 0x600; }
    if ((bcd & 0xF000) == 0xF000) { bcd -= 0x6000; }
    outBCD = bcd ;
    return outBCD ;
 }
. . .
```

Additionnner

L'addition binaire doit être suivie d'un test chiffre par chiffre en commencant par les poids faibles. Si le résultat pour le digit est supérieur à 9, il faut ajouter 6 avant de tester le digit suivant.

```
unsigned int AddBCD  (unsigned int bcd,efg) {
   unsigned int outBCD;
    bcd += efg ;  // bcd+efg → bcd
   if ((bcd & 0x000F) > 0x9) { bcd += 0x6; }
   if ((bcd & 0x00F0) > 0x90) { bcd += 0x60; }
   if ((bcd & 0x0F00) > 0x900) { bcd += 0x600; }
   if ((bcd & 0xF000) > 0x9000) { bcd += 0x6000; }
   outBCD = bcd ;
   return outBCD ;
}
```

Soustraire

La soustraction binaire doit être suivie du test chiffre par chiffre en commencant par les poids faibles. Si le résultat pour le digit est supérieur à 9, il faut soustraire 6 avant de tester le digit suivant. Si le résultat est négatif ..?

```
unsigned int SubBCD  (unsigned int bcd,efg) {
   unsigned int outBCD;
   bcd -= efg ;  // bcd-efg → bcd
   if ((bcd & 0x000F) > 0x9) { bcd -= 0x6; }
   if ((bcd & 0x00F0) > 0x90) { bcd -= 0x60; }
   if ((bcd & 00xF00) > 0x900) { bcd -= 0x600; }
   if ((bcd & 0xF000) > 0x9000) { bcd +-= 0x6000; }
   outBCD = bcd ;
   return outBCD ;
}
```

Convertir de BCD en binaire

Le plus rapide est d'avoir une table des poids binaire des bits des chiffres BCD et d'additionner les poids des bits à 1.

```
unsigned int BCD2Bin (unsigned int bcd) {
   unsigned int bin=0;
   weightsBCD2bin [16] {8000,4000,2000,1000,800,400,200,100,
                        80,40,20,10,8,4,2,1};
                    // ces poids sont convertis en binaire par le compilateur
   for  (int i=0; i<16; i++) {
     if(bcd & 0x80) {
       bin += weightsBCD2bin [i];
       bcd <<= 1 ;
       }
   }
   return bin ;
}
```

Une autre solution utilise les poids des digits (1000=0x3E8, 100=0x64,10=0xA 1=1) et un décalage de 4. Les poids sont multipliés par le digit, ce qui n'est rapide que si le processeur a une multipléication 16 bits câblée.

Convertir de binaire en BCD

Une solution est de calculer en BCD et d'ajouter les poids binaires converti en BCD (1 2 4 8 16 32…

C'est probablement plus efficace d'écrire la fonction qui multiplie par 2 un nombre BCD (légère variante de l'addition). En commençant par les poids fort, on double et ajoute le bit suivant

```
unsigned int Bin2BCD (unsigned int bin) {
   unsigned int BCD=0;
   if(bcd & 0x80) { bin=1; }
   for  (int i=0; i=<16; i++) {
     bin <<=1
     if(bin & 0x80) {double += weightsBCD2bin [i];
     bcd <<= 1 ;
   }
  return bin ;
}
```