影像壓縮軟體開發實驗

課程：嵌入式系統與軟體工程
開發學校：中央大學資工系
陳慶瀚

影像壓縮—JPEG

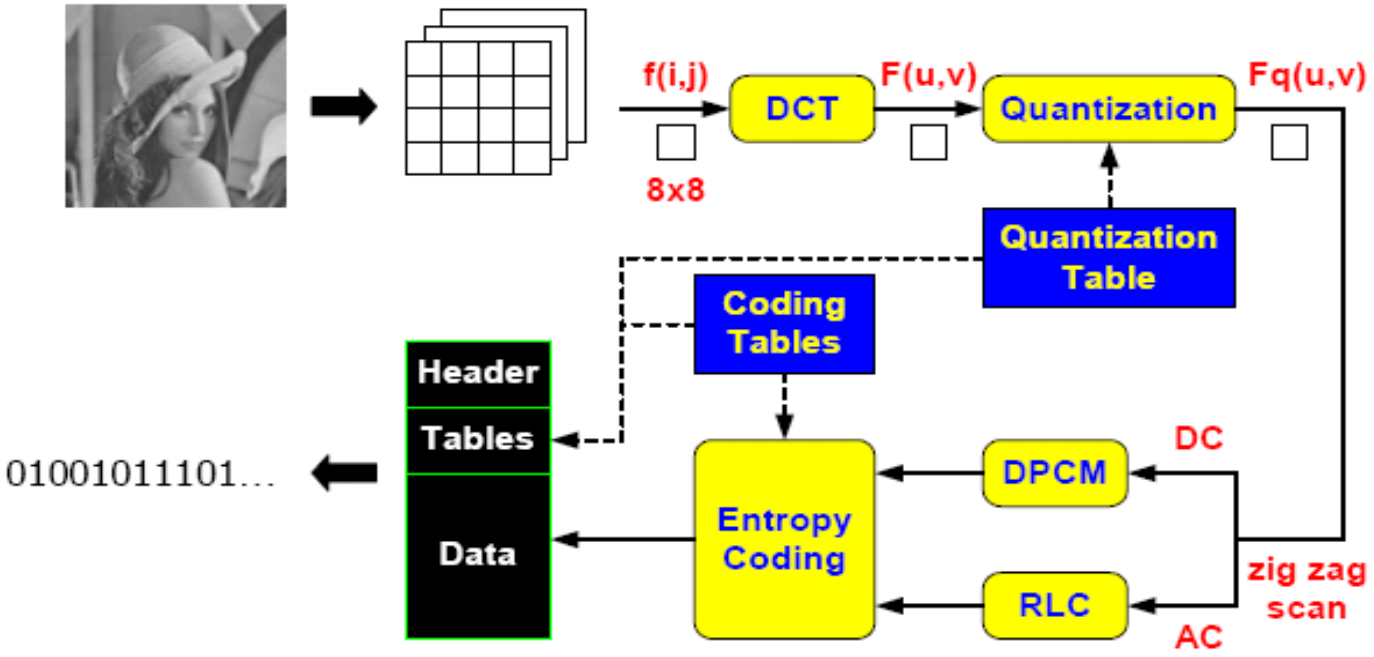
- JPEG是一種影像壓縮的格式，也是目前用途最廣泛的一種壓縮程式，其副檔名就是常見的（ . JPG ）。

JPEG之十八般武藝

- 擅長處理灰階圖形、色澤細密連續、具漸層感的圖形、具漸層性或朦朧效果的圖形點綴邊飾、用色數量遠超過 256 色的圖檔、類似相片般精細複雜的圖形。

JPEG	原始檔案	壓縮60%	壓縮20%
大小	19,464 Byte	12,035 Byte	4,227 Byte
			

實作JPEG的流程


f(i,j)
8x8
DCT
F(u,v)
Quantization
Fq(u,v)
Quantization Table
Coding Tables
Header
Tables
Data
01001011101...
Entropy Coding
DPCM
DC
RLC
AC
zig zag scan

實作JPEG的流程(cont')

- 分成下列部分
- 1. 色系變換
- 2. 離散餘弦轉換
- 3. 量化
- 4. ZigZag編碼
- 5. RLE編碼
- 6. Huffman編碼

接下來就來細細說明！

Let’s GO!

第零步

- 首先本次實驗輸入的部分，已經事先將一簡單圖檔轉換成陣列存在程式中，其中大小只有　8*8 = 64 的陣列大小

第一步　色彩轉換

- 也就是把平常用來表示顏色的RGB格式，轉為亮度以及色度的表現方法，本次實驗是用CCIR601的格式，也就是Y、Cb、Cr格式，Y代表亮度，Cb、Cr則是代表色度(藍色以及紅色的色度差)，而一般的轉換公式如下：

$$Y = 0.299R + 0.587G + 0.114B$$

$$Cb = 0.1687R - 0.3313G + 0.5B$$

$$Cr = 0.5R - 0.4187G - 0.0813B$$

色彩轉換(程式碼)

```
/*============================== RGB-YUV ==============================*/
void rgb_to_yuv(float r[8][8],float g[8][8],float b[8][8],float y[8][8],float u[8][8],float v[8][8])
{
int i,j;

for ( i=0 ; i<8 ; i++ ){
  for ( j=0 ; j<8 ; j++ ){
    y[i][j]= 0.299 *r[i][j]+0.587 *g[i][j]+0.114 *b[i][j]      ;
    u[i][j]=-0.1687*r[i][j]-0.3313*g[i][j]+0.5   *b[i][j]+128;
    v[i][j]= 0.5   *r[i][j]-0.4187*g[i][j]-0.0813*b[i][j]+128;
   }
 }
}
/*============================ RGB-YUV-End ============================*/
```

第二步　離散餘弦轉換(DCT)

- 轉化為Y、Cb、Cr的格式，圖像檔案仍然以圖點的格式儲存，因此要合併鄰近的點，這時就必須透過離散餘弦轉換，將圖點儲存的方式轉為”變化率”的儲存方式，不過這裡就是JPEG造成圖像檔案失真的地方所在，數位化時所定的係數決定了資料流失量的多寡，以及影像品質的好壞。
- 公式：

$$F(u,v)=\frac{2C(u)C(v)}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1}f(x,y)\cos\frac{(2x+1)u\pi}{2N}\cos\frac{(2y+1)v\pi}{2N}$$

離散餘弦轉換(程式碼)

```
/*================================== DCT ==================================*/
void dct (float pic_in[8][8], float enc_out[8][8])
{
 int        u,v,x,y;
 float u_cs,v_cs,Pi;

 Pi=3.1415927;
   for ( u=0 ; u<8 ; u++ ){
     for ( v=0 ; v<8 ; v++ ){
       for ( x=0 ; x<8 ; x++ ){
         for ( y=0 ; y<8 ; y++ ){
          u_cs=cos(((2*x+1)*u*Pi)/16);
          if (u==0) u_cs=(1/(sqrt(2)));
          v_cs=cos(((2*y+1)*v*Pi)/16);
          if (v==0) v_cs=(1/(sqrt(2)));
          enc_out[v][u]+=0.25*pic_in[y][x]*u_cs*v_cs;
         }
       }
     }
   }
}
/*================================== DCT-End ==============================*/
```

第三步 量化

- 圖像數據轉換回頻率係數後，還要接受一項量化程序，才能進入編碼階段。量化需要兩個8*8的矩陣，一個處理亮度係數，一個處理色度係數，將頻率系數除以量化矩陣的值後，取得與商數最近的整數，我們稱作量化。目的是將頻率係數由浮點數轉變為整數，這才方便最後的編碼階段，不過因為取整數的這個步驟，也損失了一些數據內容。

量化(cont')

- 以下為本次實驗所用的量化表
- 色度量化表：

17	18	24	47	99	99	99	99
18	21	26	66	99	99	99	99
24	26	56	99	99	99	99	99
47	66	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99

- 亮度量化表：

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

量化(程式碼)

```
/*================================== Quant ================================*/
void quantize(float dctb[8][8],float qb[8][8],int n)
{
int u,v;

for ( v=0 ; v<8 ; v++ ){
  for ( u=0 ; u<8 ; u++ ){
    if (n==0)
      qb[v][u]=dctb[v][u]/q0[v][u];
    else
      qb[v][u]=dctb[v][u]/q1[v][u];
  }
}

}
/*================================ Quant-End ==============================*/
```

第四步 ZigZag編碼

- 量化後的所有數據都是線性存放的，如果我們一行行處理這64個數字，每行的結尾的點和下行開始的點就沒有什麼關係了，所以JPEG規定按照下表來整理這64個數：

0	1	5	6	14	15	27	28
2	4	7	13	16	26	29	42
3	8	12	17	25	30	41	43
9	11	18	24	31	40	44	53
10	19	23	32	39	45	52	54
20	22	33	38	46	51	55	60
21	34	37	47	50	56	59	61
35	36	48	49	57	58	68	63

ZigZag(程式碼)

```
/*================================== ZigZag ================================*/
void zigzag(float quant_in[8][8], float zigzaged_out[64])
{
int i;
int u[64]={1,2,1,1,2,3,4,3,2,1,1,2,3,4,5,6,
           5,4,3,2,1,1,2,3,4,5,6,7,8,7,6,5,
           4,3,2,1,2,3,4,5,6,7,8,8,7,6,5,4,
           3,4,5,6,7,8,8,7,6,5,6,7,8,8,7,8
          };
int v[64]={1,1,2,3,2,1,1,2,3,4,5,4,3,2,1,1,
           2,3,4,5,6,7,6,5,4,3,2,1,1,2,3,4,
           5,6,7,8,8,7,6,5,4,3,2,3,4,5,6,7,
           8,8,7,6,5,4,5,6,7,8,8,7,6,7,8,8
          };

  for ( i=0 ; i<64 ; i++){
    zigzaged_out[i]=quant_in[u[i]][v[i]];
  }
}
/*================================ ZigZag-End ==============================*/
```

第五步 RLE 編碼

- 64個變換數經量化後，左上角係數是直流分量（DC係數），即空間域中64個圖像採樣值的均值。相鄰8*8塊之間的DC係數一般有很強的相關性，JPEG標準對DC係數採用DPCM（差分脈衝碼調制）方法，即對相鄰區塊之間的L係數的差值進行編碼。其餘63個交流分量（AC係數）使用游程編碼，從左上角開始沿對角線方向，以Z字型（zigzag）進行掃描直到結束。

RLE 編碼 (cont')

- 1、DC係數編碼

  使用DPCM對直流係數進行編碼，8*8圖像塊經過 DCT編碼之後得到的 DC係數有兩個特點：一是係數的數值比較大；二是相鄰的8*8圖像塊的DC系數值變化不大。根據這個特點，JPEG算法使用了DPCM技術，對相鄰圖像塊之間量化DC係數的差值（Diff）進行編碼。

  公式： Diff=DC（i）- DC（i-1）

RLE 編碼 (cont')

- DC係數編碼表

差值位數	DC 差值內容
0	0
1	-1，1
2	-3，-2，2，3
3	-7，…，-4，4，…，7
4	-15，…，-8，8，…，15
5	-31，…，-16，16，…，31
6	-63，…，-32，32，…，63
7	-127，…，-64，64，…，127
8	-255，…，-128，128，…，255
9	-511，…，-256，256，…，511
10	-1023，…，-512，512，…，1023
11	-2047，…，-1024，1024，…，2047

RLE 編碼 (cont')

2、AC係數編碼

使用RLE對AC係數進行編碼。量化AC係數的特點是1-64向量中包含有許多“0”係數，並且許多的“0”係數都是連續的，因此使用非常簡單和直觀的RLE編碼對它們編碼。

RLE 編碼 (cont')

- **AC係數編碼 表**

数值	SSSS组	实际保存值
0	0	不保存
-1, 1	1	0, 1
-3, -2, 2, 3	2	00, 01, 10, 11
-7, -6, -5, -4, 4, 5, 6, 7	3	000, 001, 010, 011, 100, 101, 110, 111
-15, . . , -8, 8, . . , 15	4	0000, . . , 0111, 1000, . . , 1111
-31, . . , -16, 16, . . , 31	5	00000, . . , 01111, 10000, . . , 11111
-63, . . , -32, 32, . . , 63	6	. .
-127, . . , -64, 64, . . , 127	7	. .
-255, . . , -128, 128, . . , 255	8	. .
-511, . . , -256, 256, . . , 511	9	. .
-1023, . . , -512, 512, . . , 1023	10	. .
-2047, . . , -1024, 1024, . . , 2047	11	. .
-4095, . . , -2048, 2048, . . , 4095	12	. .
-8191, . . , -4096, 4096, . . , 8191	13	. .
-16383, . . , -8192, 8192, . . , 16383	14	. .
-32767, . . , -16384, 16384, . . , 32767	15	. .

RLE 編碼 (cont')

範例：

現在向量中有許多連續的“0”，可以使用RLE來壓縮掉這些“0”。跳過第一個DC向量，假設有一組AC向量（64個的後63個）是：57,45,0,0,0,0,23,0,-30,-16,0,0,1,0,0,0,0,0,0,0,..,0

RLE 編碼 (cont')

經過RLE壓縮後如下：

(0,57);(0,45);(4,23);(1,-30);(0,-16);(2,1);EOB。

EOB是一個結束標記，表示後面都是“0”了。實際上我們用(0,0)表示EOB。但是，如果這組數字不以0結束，那麼就不需要EOB。零行程長度超過15個時，有一個符號(15，0)。

第六步　Huffman 編碼

- Huffman編碼器可以使用查表方法進行編碼。壓縮資料符號時，Huffman編碼器對出現 頻率較高的符號分配比較短的代碼，而對出現頻率比較高低的符號分配比較長的代碼。這種可變長度的Huffman表可以事先定義。
- 編碼時，每個矩陣資料的DC值與63個AC值，將分別使用不同的Huffman編碼表，而亮度與色度也需要不同的Huffman編碼表，所以一共需要四個編碼表，才能順利地完成JPEG編碼工作。

Huffman 編碼(DC亮度&色度編碼表)

差值位数	编码位数	Huffman编码
亮度DC差值的Huffman编码表		
0	2	00
1	3	010
2	3	011
3	3	100
4	3	101
5	3	110
6	4	1110
7	5	11110
8	6	111110
9	7	1111110
10	8	11111110
11	9	111111110
色度DC差值的Huffman编码表		
0	2	00
1	2	01
2	2	10
3	3	110
4	4	1110
5	5	11110
6	6	111110
7	7	1111110
8	8	11111110
9	9	111111110
10	10	1111111110
11	11	11111111110

Huffman 編碼(AC亮度編碼表-1/4)

Run/Size	Code length	Code word
0/0 (EOB)	4	1010
0/1	2	00
0/2	2	01
0/3	3	100
0/4	4	1011
0/5	5	11010
0/6	7	1111000
0/7	8	11111000
0/8	10	1111110110
0/9	16	1111111110000010
0/A	16	1111111110000011
1/1	4	1100
1/2	5	11011
1/3	7	1111001
1/4	9	111110110
1/5	11	11111110110
1/6	16	1111111110000100
1/7	16	1111111110000101
1/8	16	1111111110000110
1/9	16	1111111110000111
1/A	16	1111111110001000
2/1	5	11100
2/2	8	11111001
2/3	10	1111110111
2/4	12	111111110100
2/5	16	1111111110001001
2/6	16	1111111110001010
2/7	16	1111111110001011
2/8	16	1111111110001100
2/9	16	1111111110001101
2/A	16	1111111110001110
3/1	6	111010
3/2	9	111110111
3/3	12	111111110101
3/4	16	1111111110001111
3/5	16	1111111110010000
3/6	16	1111111110010001
3/7	16	1111111110010010
3/8	16	1111111110010011
3/9	16	1111111110010100
3/A	16	1111111110010101

Huffman 編碼(AC亮度編碼表-2/4)

Run/Size	Code length	Code word
4/1	6	111011
4/2	10	1111111000
4/3	16	1111111110010110
4/4	16	1111111110010111
4/5	16	1111111110011000
4/6	16	1111111110011001
4/7	16	1111111110011010
4/8	16	1111111110011011
4/9	16	1111111110011100
4/A	16	1111111110011101
5/1	7	1111010
5/2	11	11111110111
5/3	16	1111111110011110
5/4	16	1111111110011111
5/5	16	1111111110100000
5/6	16	1111111110100001
5/7	16	1111111110100010
5/8	16	1111111110100011
5/9	16	1111111110100100
5/A	16	1111111110100101
6/1	7	1111011
6/2	12	111111110110
6/3	16	1111111110100110
6/4	16	1111111110100111
6/5	16	1111111110101000
6/6	16	1111111110101001
6/7	16	1111111110101010
6/8	16	1111111110101011
6/9	16	1111111110101100
6/A	16	1111111110101101
7/1	8	11111010
7/2	12	111111110111
7/3	16	1111111110101110
7/4	16	1111111110101111
7/5	16	1111111110110000
7/6	16	1111111110110001
7/7	16	1111111110110010
7/8	16	1111111110110011
7/9	16	1111111110110100
7/A	16	1111111110110101
8/1	9	111111000
8/2	15	111111111000000

Huffman 編碼(AC亮度編碼表-3/4)

Run/Size	Code length	Code word
8/3	16	1111111111010110
8/4	16	1111111111010111
8/5	16	1111111111011000
8/6	16	1111111111011001
8/7	16	1111111111011010
8/8	16	1111111111011011
8/9	16	1111111111011100
8/A	16	1111111111011101
9/1	9	111111001
9/2	16	1111111111011110
9/3	16	1111111111011111
9/4	16	1111111111000000
9/5	16	1111111111000001
9/6	16	1111111111000010
9/7	16	1111111111000011
9/8	16	1111111111000100
9/9	16	1111111111000101
9/A	16	1111111111000110
A/1	9	111111010
A/2	16	1111111111000111
A/3	16	1111111111001000
A/4	16	1111111111001001
A/5	16	1111111111001010
A/6	16	1111111111001011
A/7	16	1111111111001100
A/8	16	1111111111001101
A/9	16	1111111111001110
A/A	16	1111111111001111
B/1	10	1111111001
B/2	16	1111111111010000
B/3	16	1111111111010001
B/4	16	1111111111010010
B/5	16	1111111111010011
B/6	16	1111111111010100
B/7	16	1111111111010101
B/8	16	1111111111010110
B/9	16	1111111111010111
B/A	16	1111111111011000
C/1	10	1111111010
C/2	16	1111111111011001
C/3	16	1111111111011010
C/4	16	1111111111011011

Huffman 編碼(AC亮度編碼表-4/4)

Run/Size	Code length	Code word
C/5	16	1111111111011100
C/6	16	1111111111011101
C/7	16	1111111111011110
C/8	16	1111111111011111
C/9	16	1111111111100000
C/A	16	1111111111100001
D/1	11	11111111000
D/2	16	1111111111100010
D/3	16	1111111111100011
D/4	16	1111111111100100
D/5	16	1111111111100101
D/6	16	1111111111100110
D/7	16	1111111111100111
D/8	16	1111111111101000
D/9	16	1111111111101001
D/A	16	1111111111101010
E/1	16	1111111111101011
E/2	16	1111111111101100
E/3	16	1111111111101101
E/4	16	1111111111101110
E/5	16	1111111111101111
E/6	16	1111111111110000
E/7	16	1111111111110001
E/8	16	1111111111110010
E/9	16	1111111111110011
E/A	16	1111111111110100
F/0 (ZRL)	11	11111111001
F/1	16	1111111111110101
F/2	16	1111111111110110
F/3	16	1111111111110111
F/4	16	1111111111111000
F/5	16	1111111111111001
F/6	16	1111111111111010
F/7	16	1111111111111011
F/8	16	1111111111111100
F/9	16	1111111111111101
F/A	16	1111111111111110

Huffman 編碼(AC色度編碼表-1/4)

Run/Size	Code length	Code word
0/0 (EOB)	2	00
0/1	2	01
0/2	3	100
0/3	4	1010
0/4	5	11000
0/5	5	11001
0/6	6	111000
0/7	7	1111000
0/8	9	111110100
0/9	10	1111110110
0/A	12	111111110100
1/1	4	1011
1/2	6	111001
1/3	8	11110110
1/4	9	111110101
1/5	11	11111110110
1/6	12	111111110101
1/7	16	1111111110001000
1/8	16	1111111110001001
1/9	16	1111111110001010
1/A	16	1111111110001011
2/1	5	11010
2/2	8	11110111
2/3	10	1111110111
2/4	12	111111110110
2/5	15	111111111000010
2/6	16	1111111110001100
2/7	16	1111111110001101
2/8	16	1111111110001110
2/9	16	1111111110001111
2/A	16	1111111110010000
3/1	5	11011
3/2	8	11111000
3/3	10	1111111000
3/4	12	111111110111
3/5	16	1111111110010001
3/6	16	1111111110010010
3/7	16	1111111110010011
3/8	16	1111111110010100
3/9	16	1111111110010101
3/A	16	1111111110010110
4/1	6	111010

Huffman 編碼(AC色度編碼表-2/4)

Run/Size	Code length	Code word
4/2	9	111110110
4/3	16	1111111110010111
4/4	16	1111111110011000
4/5	16	1111111110011001
4/6	16	1111111110011010
4/7	16	1111111110011011
4/8	16	1111111110011100
4/9	16	1111111110011101
4/A	16	1111111110011110
5/1	6	111011
5/2	10	1111111001
5/3	16	1111111110011111
5/4	16	1111111110100000
5/5	16	1111111110100001
5/6	16	1111111110100010
5/7	16	1111111110100011
5/8	16	1111111110100100
5/9	16	1111111110100101
5/A	16	1111111110100110
6/1	7	1111001
6/2	11	11111110111
6/3	16	1111111110100111
6/4	16	1111111110101000
6/5	16	1111111110101001
6/6	16	1111111110101010
6/7	16	1111111110101011
6/8	16	1111111110101100
6/9	16	1111111110101101
6/A	16	1111111110101110
7/1	7	1111010
7/2	11	11111111000
7/3	16	1111111110101111
7/4	16	1111111110110000
7/5	16	1111111110110001
7/6	16	1111111110110010
7/7	16	1111111110110011
7/8	16	1111111110110100
7/9	16	1111111110110101
7/A	16	1111111110110110
8/1	8	11111001
8/2	16	1111111110110111
8/3	16	1111111110111000

Huffman 編碼(AC色度編碼表-3/4)

Run/Size	Code length	Code word
8/4	16	1111111110111001
8/5	16	1111111110111010
8/6	16	1111111110111011
8/7	16	1111111110111100
8/8	16	1111111110111101
8/9	16	1111111110111110
8/A	16	1111111110111111
9/1	9	111110111
9/2	16	1111111111000000
9/3	16	1111111111000001
9/4	16	1111111111000010
9/5	16	1111111111000011
9/6	16	1111111111000100
9/7	16	1111111111000101
9/8	16	1111111111000110
9/9	16	1111111111000111
9/A	16	1111111111001000
A/1	9	111111000
A/2	16	1111111111001001
A/3	16	1111111111001010
A/4	16	1111111111001011
A/5	16	1111111111001100
A/6	16	1111111111001101
A/7	16	1111111111001110
A/8	16	1111111111001111
A/9	16	1111111111010000
A/A	16	1111111111010001
B/1	9	111111001
B/2	16	1111111111010010
B/3	16	1111111111010011
B/4	16	1111111111010100
B/5	16	1111111111010101
B/6	16	1111111111010110
B/7	16	1111111111010111
B/8	16	1111111111011000
B/9	16	1111111111011001
B/A	16	1111111111011010
C/1	9	111111010
C/2	16	1111111111011011
C/3	16	1111111111011100
C/4	16	1111111111011101
C/5	16	1111111111011110

Huffman 編碼(AC色度編碼表-4/4)

Run/Size	Code length	Code word
C/6	16	1111111111011111
C/7	16	1111111111100000
C/8	16	1111111111100001
C/9	16	1111111111100010
C/A	16	1111111111100011
D/1	11	11111111001
D/2	16	1111111111100100
D/3	16	1111111111100101
D/4	16	1111111111100110
D/5	16	1111111111100111
D/6	16	1111111111101000
D/7	16	1111111111101001
D/8	16	1111111111101010
D/9	16	1111111111101011
D/A	16	1111111111101100
E/1	14	11111111100000
E/2	16	1111111111101101
E/3	16	1111111111101110
E/4	16	1111111111101111
E/5	16	1111111111110000
E/6	16	1111111111110001
E/7	16	1111111111110010
E/8	16	1111111111110011
E/9	16	1111111111110100
E/A	16	1111111111110101
F/0 (ZRL)	10	1111111010
F/1	15	111111111000011
F/2	16	1111111111110110
F/3	16	1111111111110111
F/4	16	1111111111111000
F/5	16	1111111111111001
F/6	16	1111111111111010
F/7	16	1111111111111011
F/8	16	1111111111111100
F/9	16	1111111111111101
F/A	16	1111111111111110

接下來，要實作在哪呢？

- 期待實驗5-2…