WS28 lib for 8MHz ATtinys

There are two timings very critical with the WS2812 B and WS2813B RGB chips. They are adjusted with NOP and minimum quartz value for AVR microcontrollers is 8MHz, maybe 7. The same function probably works also at 10 MHz.
It is of course important not to be interrupted during the data transfer (Arduino delay() starts automatically the interrupts) and the time to prepare and call the next pixel must be below 40 microseconds (to be checked).

```
  //WS28.h lib
  typedef uint8_t byte;
  #define bitSet(x,y) x|=(1<<y)
  #define bitClear(x,y) x&=~(1<<y)

  #define bP 2 // RB2  do not use pinMode
  #define POn  bitSet   (PORTB,bP)
  #define POff bitClear (PORTB,bP)
  void SetupWS2813 () { bitSet (DDRB,bP); }
```

8MHz instructions (ATtiny24 25 etc)	8MHz instructions (AtMega328)

`// ne pas utiliser delay()`
`#define Calib1ms 470   // 1ms  8 MHz`
`void delMs (int dm) {`
`   for (volatile int i=0; i<dm;`
`i++) {`
`     }`
`}`
`#define nop asm ("nop")`
`void S8 ( byte dd ) {`
`  volatile byte cc=0;`
`  while (cc++ < 8) {`
`    POn; //nop;  // 0  nop at 8 MHz`
`    if (!(dd&0x80)) { POff;}`
`    nop; nop; POff;    //  2 nop`
`    dd <<=1;`
`  }`
`}`	

```
void LinRGB( byte rr, byte gg, byte bb ) {
  S8(gg); S8(rr); S8(bb);
}
void Close() {  // delay 350 us
 for (volatile int j=0; j<300; j++) {}
}
void Clear() {
  for (byte i=0;i<Npix;i++) {
     LinRGB (0,0,0);
  }
  Close();
}
byte taConv [32]={0,2,3,4, 5,6,7,8, \
           9,10,11,12, 14,16,18,20, \
          23,27,32,37, 43,51,62,68, \
    83,100,120,144,168, 198,224,255};
byte Conv (byte vv) {
  return (taConv [vv/8]);
}
void LogRGB( byte rr, byte gg, byte bb ) {
  S8(taConv [gg/8]); S8(taConv [rr/8]); S8(taConv [bb/8]);
}
```

See www.didel.com/WS28Lib.pdf for comments

jdn 170630