Discussion1

(1) Set correct value

```
self.scale = 0.92 # convert bit value(LSB) to gauss. DigitalResolution

# Configuration Register A
self.write_byte(HMC5883L_CRA, 0b01110000)
# CRA6-CRA5 = 11 -> 8 samples per measurement
# CRA4-CRA2 = 100 -> Data Output Rate = 15Hz
# CRA1-CRA0 = 00 -> Normal measurement configuration

# Configuration Register B
self.write_byte(HMC5883L_CRB, 0b00100000)
# CRA7-CRB5 = 001 Gain = 1090 (LSB/Gauss)

# Mode Register
self.write_byte(HMC5883L_MR, 0b00000000)
# MR1-MR0 = 00 Continuous-Measurement Mode
```

(2) loop

```
467     while True:
468         tempC = barometer.getTempC()
469         tempF = barometer.getTempF()
470         press = barometer.getPress()
471         altitude = barometer.getAltitude()
472         magx = compass.getX()
473         magy = compass.getY()
474         magz = compass.getZ()
475
476         # -------------------------------------------------
477         # calculate pitch, roll, tilt
478         aX = adxl345.getX()
479         aY = adxl345.getY()
480         aZ = adxl345.getZ()
481
482         roll = (roll1+gyro.getYangle())*0.98 + atan2(aY,aZ) * 0.02
483         pitch = (pitch1+gyro.getXangle())*0.98 + -1 * atan2(-aX,sqrt(aY*aY+aZ*aZ)) * 0.02
484         # -------------------------------------------------
485         print ("pitch = %.3f" % (pitch))
486         print ("roll = %.3f" % (roll))
487
488         roll1=atan2(aY,aZ)
489         pitch1=-1 * atan2(-aX,sqrt(aY*aY+aZ*aZ))
490
491         # -------------------------------------------------
492         # Heading
493         bearing1  = degrees(atan2(magy, magx))
494
495         if (bearing1 < 0):
496             bearing1 += 360
497         if (bearing1 > 360):
498             bearing1 -= 360
499         bearing1 = bearing1 + compass.angle_offset
500
501         # Tilt compensate
502         compx = magx * cos(pitch) + magz * sin(pitch)
503         compy = magx * sin(roll) * sin(pitch) \
504                 + magy * cos(roll) \
505                 - magz * sin(roll) * cos(pitch)
506
507         bearing2  = degrees(atan2(compy, compx))
```

```
507         bearing2  = degrees(atan2(compy, compx))
508         if (bearing2 < 0):
509             bearing2 += 360
510         if (bearing2 > 360):
511             bearing2 -= 360
512         bearing2 = bearing2 + compass.angle_offset
513         # ----------------------------------------------------
514
515
516 #        print ("Compass: " )
517 #        print ("X = %d ," % ( magx )),
518 #        print ("Y = %d ," % ( magy )),
519 #        print ("Z = %d (gauss)" % ( magz ))
520 #        print ("tiltX = %.3f ," % ( compx )),
521 #        print ("tiltY = %.3f ," % ( compy )),
522
523
524 #        print ("Angle offset = %.3f deg" % ( compass.angle_offset ))
525 #        print ("Original Heading = %.3f deg, " % ( bearing1 )),
526         print ("Tilt Heading = %.3f deg, " % ( bearing2 ))
527         print ("Altitude: %f m s.l.m" %(altitude))
528         print ("")
529         time.sleep(1)
```

(3) calibrate

```
class ADXL345(IMU):

    ADDRESS = ADXL345_ADDRESS

    def __init__(self) :
        #Class Properties
        self.Xoffset = -4.0  # unit: G, modify by yourself
        self.Yoffset = -222.0  # unit: G, modify by yourself
        self.Zoffset = 10.0  # unit: G, modify by yourself
```

Discussion2

(4) Set correct value

```
def getTempC(self) :
    # print ("Calculating temperature...")
    self.write_byte(0xF4, 0x2E)
    time.sleep(0.005)

    ut = self.read_word(0xF6,0)

    # calculate true temperature
    x1 = ((ut - self.ac6_val) * self.ac5_val) >> 15
    x2 = (self.mc_val << 11) // (x1 + self.md_val)
    b5 = x1 + x2
    self.tempC = ((b5 + 8) >> 4) / 10.0

    return self.tempC
```

```
# read uncompensated pressure value
def getPress(self) :
    # print ("Calculating temperature...")
    self.write_byte(0xF4, 0x2E)
    time.sleep(0.005)

    ut = self.read_word(0xF6,0)

    x1 = ((ut - self.ac6_val) * self.ac5_val) >> 15
    x2 = (self.mc_val << 11) // (x1 + self.md_val)
    b5 = x1 + x2

    #print ("Calculating pressure...")
    self.write_byte(0xF4, 0x34 + (self.oversampling << 6))
    time.sleep(0.04)
```

(2) loop

```
sensors = gy801()

barometer = sensors.baro
while True:
    tempC = barometer.getTempC()
    tempF = barometer.getTempF()
    press = barometer.getPress()
    altitude = barometer.getAltitude()

    print ("Barometer:" )
    print ("   Temp: %f C (%f F)" %(tempC,tempF))
    print ("   Press: %f (hPa)" %(press))
    print ("   Altitude: %f m s.l.m" %(altitude))
    time.sleep(1)
```