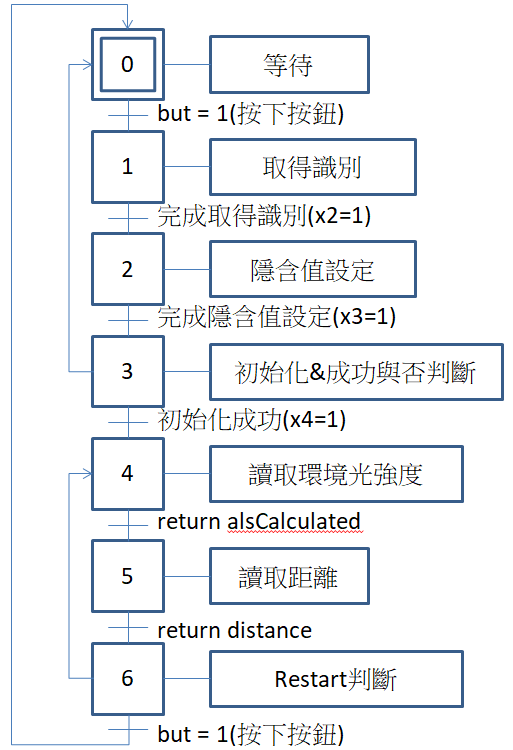
嵌入式系統期中報告

醫工四10311001王思敏

1. 腳位:

|  |  |
| --- | --- |
| Nucleo-F767ZI | VL6180X |
| 3.3V | Vin |
| gnd | gnd |
| D14 | sda |
| D15 | scl |

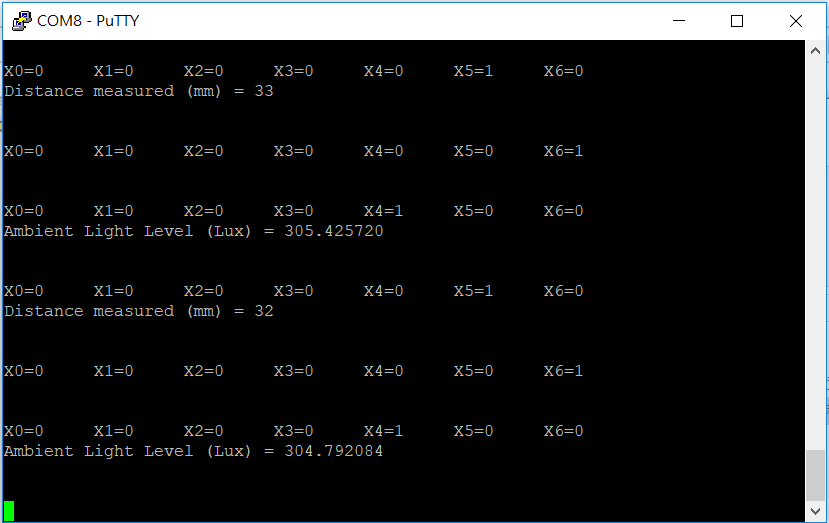
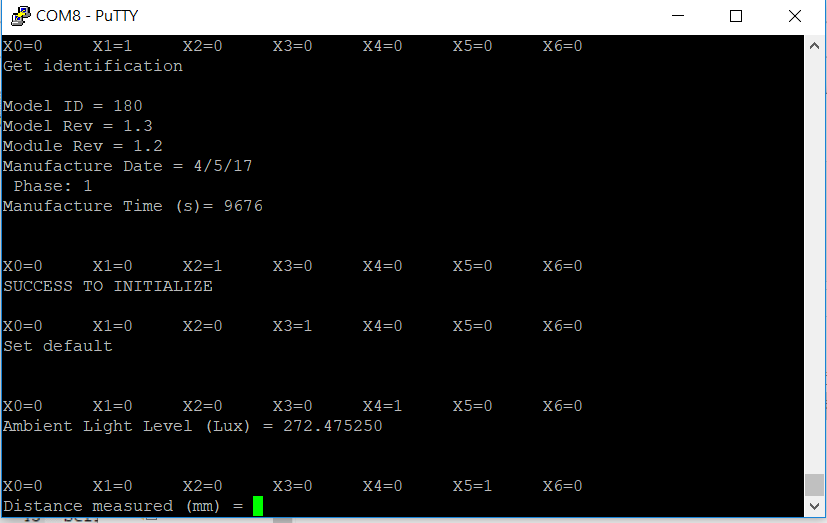
1. Grafcet:



1. API定義:

|  |  |  |
| --- | --- | --- |
| 功能 | API | 參數 |
| 取得識別 | getIdentification | struct VL6180xIdentification \*temp |
| 設預設值 | VL6180xDefautSettings | null |
| 初始化 | VL6180xInit | null |
| 讀取環境光強 | getAmbientLight | alsCalculated |
| 讀取距離 | getDistance | distance |

1. 結果:



1. 程式碼:

#include "mbed.h"

#include "VL6180x.h"

#define VL6180X\_ADDRESS 0x29

int x0, x1, x2, x3, x4, x5, x6, but;

Timer debounceTimer;

InterruptIn myButton(USER\_BUTTON);

VL6180x::VL6180x(PinName sda, PinName scl, uint8\_t addr) : m\_i2c(sda, scl), m\_addr(addr) {}

VL6180xIdentification identification;

Serial pc(USBTX, USBRX);

VL6180x sensor(D14, D15, VL6180X\_ADDRESS<<1);

int VL6180x::VL6180xInit(void){

uint8\_t data; //for temp data storage

data = VL6180x\_getRegister(VL6180X\_SYSTEM\_FRESH\_OUT\_OF\_RESET);

wait\_ms(50);

if(data != 1) return VL6180x\_FAILURE\_RESET;

VL6180x\_setRegister(0x0207, 0x01);

VL6180x\_setRegister(0x0208, 0x01);

VL6180x\_setRegister(0x0096, 0x00);

VL6180x\_setRegister(0x0097, 0xfd);

VL6180x\_setRegister(0x00e3, 0x00);

VL6180x\_setRegister(0x00e4, 0x04);

VL6180x\_setRegister(0x00e5, 0x02);

VL6180x\_setRegister(0x00e6, 0x01);

VL6180x\_setRegister(0x00e7, 0x03);

VL6180x\_setRegister(0x00f5, 0x02);

VL6180x\_setRegister(0x00d9, 0x05);

VL6180x\_setRegister(0x00db, 0xce);

VL6180x\_setRegister(0x00dc, 0x03);

VL6180x\_setRegister(0x00dd, 0xf8);

VL6180x\_setRegister(0x009f, 0x00);

VL6180x\_setRegister(0x00a3, 0x3c);

VL6180x\_setRegister(0x00b7, 0x00);

VL6180x\_setRegister(0x00bb, 0x3c);

VL6180x\_setRegister(0x00b2, 0x09);

VL6180x\_setRegister(0x00ca, 0x09);

VL6180x\_setRegister(0x0198, 0x01);

VL6180x\_setRegister(0x01b0, 0x17);

VL6180x\_setRegister(0x01ad, 0x00);

VL6180x\_setRegister(0x00ff, 0x05);

VL6180x\_setRegister(0x0100, 0x05);

VL6180x\_setRegister(0x0199, 0x05);

VL6180x\_setRegister(0x01a6, 0x1b);

VL6180x\_setRegister(0x01ac, 0x3e);

VL6180x\_setRegister(0x01a7, 0x1f);

VL6180x\_setRegister(0x0030, 0x00);

return 0;

}

VL6180x::~VL6180x(void) {

};

void VL6180x::VL6180xDefautSettings(void){

VL6180x\_setRegister(VL6180X\_SYSTEM\_INTERRUPT\_CONFIG\_GPIO, (4 <<

3)|(4) );

VL6180x\_setRegister(VL6180X\_SYSTEM\_MODE\_GPIO1, 0x10); VL6180x\_setRegister(VL6180X\_READOUT\_AVERAGING\_SAMPLE\_PERIOD, 0x30);

VL6180x\_setRegister(VL6180X\_SYSALS\_ANALOGUE\_GAIN, 0x46); VL6180x\_setRegister(VL6180X\_SYSRANGE\_VHV\_REPEAT\_RATE, 0xFF); VL6180x\_setRegister(VL6180X\_SYSALS\_INTEGRATION\_PERIOD, 0x63); VL6180x\_setRegister(VL6180X\_SYSRANGE\_VHV\_RECALIBRATE, 0x01);

VL6180x\_setRegister(VL6180X\_SYSRANGE\_INTERMEASUREMENT\_PERIOD,

0x09);

VL6180x\_setRegister(VL6180X\_SYSALS\_INTERMEASUREMENT\_PERIOD, 0x0A);

VL6180x\_setRegister(VL6180X\_SYSTEM\_INTERRUPT\_CONFIG\_GPIO, 0x24);

VL6180x\_setRegister(VL6180X\_SYSRANGE\_MAX\_CONVERGENCE\_TIME, 0x32);

VL6180x\_setRegister(VL6180X\_SYSRANGE\_RANGE\_CHECK\_ENABLES, 0x10 |

0x01); VL6180x\_setRegister16bit(VL6180X\_SYSRANGE\_EARLY\_CONVERGENCE\_ESTIMA

TE, 0x7B );

VL6180x\_setRegister16bit(VL6180X\_SYSALS\_INTEGRATION\_PERIOD, 0x64);

VL6180x\_setRegister(VL6180X\_READOUT\_AVERAGING\_SAMPLE\_PERIOD,0x30);

VL6180x\_setRegister(VL6180X\_SYSALS\_ANALOGUE\_GAIN,0x40);

VL6180x\_setRegister(VL6180X\_FIRMWARE\_RESULT\_SCALER,0x01);

}

void VL6180x::getIdentification(struct VL6180xIdentification \*temp){

temp->idModel = VL6180x\_getRegister(VL6180X\_IDENTIFICATION\_MODEL\_ID);

temp->idModelRevMajor =

VL6180x\_getRegister(VL6180X\_IDENTIFICATION\_MODEL\_REV\_MAJOR);

temp->idModelRevMinor =

VL6180x\_getRegister(VL6180X\_IDENTIFICATION\_MODEL\_REV\_MINOR);

temp->idModuleRevMajor =

VL6180x\_getRegister(VL6180X\_IDENTIFICATION\_MODULE\_REV\_MAJOR);

temp->idModuleRevMinor =

VL6180x\_getRegister(VL6180X\_IDENTIFICATION\_MODULE\_REV\_MINOR);

temp->idDate = VL6180x\_getRegister16bit(VL6180X\_IDENTIFICATION\_DATE);

temp->idTime = VL6180x\_getRegister16bit(VL6180X\_IDENTIFICATION\_TIME);

}

uint8\_t VL6180x::changeAddress(uint8\_t old\_address, uint8\_t new\_address){

if( old\_address == new\_address) return old\_address;

if( new\_address > 127) return old\_address;

VL6180x\_setRegister(VL6180X\_I2C\_SLAVE\_DEVICE\_ADDRESS, new\_address);

m\_addr=new\_address<<1;

return VL6180x\_getRegister(VL6180X\_I2C\_SLAVE\_DEVICE\_ADDRESS);

}

uint8\_t VL6180x::getDistance() {

uint8\_t distance;

VL6180x\_setRegister(VL6180X\_SYSRANGE\_START, 0x01);

wait\_ms(10);

distance = VL6180x\_getRegister(VL6180X\_RESULT\_RANGE\_VAL);

VL6180x\_setRegister(VL6180X\_SYSTEM\_INTERRUPT\_CLEAR, 0x07);

return distance;

}

float VL6180x::getDistance\_m() {

float distance;

VL6180x\_setRegister(VL6180X\_SYSRANGE\_START, 0x01);

wait\_ms(10);

distance = 0.001\*(float)VL6180x\_getRegister(VL6180X\_RESULT\_RANGE\_VAL);

VL6180x\_setRegister(VL6180X\_SYSTEM\_INTERRUPT\_CLEAR, 0x07);

return distance;

}

float VL6180x::getAmbientLight(vl6180x\_als\_gain VL6180X\_ALS\_GAIN)

{

VL6180x\_setRegister(VL6180X\_SYSALS\_ANALOGUE\_GAIN, (0x40 |

VL6180X\_ALS\_GAIN));

VL6180x\_setRegister(VL6180X\_SYSALS\_START, 0x01);

wait\_ms(100);

VL6180x\_setRegister(VL6180X\_SYSTEM\_INTERRUPT\_CLEAR, 0x07);

unsigned int alsRaw = VL6180x\_getRegister16bit(VL6180X\_RESULT\_ALS\_VAL);

unsigned int alsIntegrationPeriodRaw =

VL6180x\_getRegister16bit(VL6180X\_SYSALS\_INTEGRATION\_PERIOD);

float alsIntegrationPeriod = 100.0 / alsIntegrationPeriodRaw ;

float alsGain = 0.0;

switch (VL6180X\_ALS\_GAIN){

case GAIN\_20: alsGain = 20.0; break;

case GAIN\_10: alsGain = 10.32; break;

case GAIN\_5: alsGain = 5.21; break;

case GAIN\_2\_5: alsGain = 2.60; break;

case GAIN\_1\_67: alsGain = 1.72; break;

case GAIN\_1\_25: alsGain = 1.28; break;

case GAIN\_1: alsGain = 1.01; break;

case GAIN\_40: alsGain = 40.0; break;

}

float alsCalculated = (float)0.32 \* ((float)alsRaw / alsGain) \*alsIntegrationPeriod;

return alsCalculated;

}

uint8\_t VL6180x::VL6180x\_getRegister(uint16\_t registerAddr)

{

uint8\_t data;

char data\_write[2];

char data\_read[1];

data\_write[0] = (registerAddr >> 8) & 0xFF;

data\_write[1] = registerAddr & 0xFF;

m\_i2c.write(m\_addr, data\_write, 2,0);

m\_i2c.read(m\_addr,data\_read,1,1);

data=data\_read[0];

return data;

}

uint16\_t VL6180x::VL6180x\_getRegister16bit(uint16\_t registerAddr)

{

uint8\_t data\_low;

uint8\_t data\_high;

uint16\_t data;

char data\_write[2];

char data\_read[2];

data\_write[0] = (registerAddr >> 8) & 0xFF;

data\_write[1] = registerAddr & 0xFF;

m\_i2c.write(m\_addr, data\_write, 2,0);

m\_i2c.read(m\_addr,data\_read,2,1);

data\_high = data\_read[0];

data\_low = data\_read[1];

data = (data\_high << 8)|data\_low;

return data;

}

void VL6180x::VL6180x\_setRegister(uint16\_t registerAddr, uint8\_t data)

{

char data\_write[3];

data\_write[0] = (registerAddr >> 8) & 0xFF;

data\_write[1] = registerAddr & 0xFF;

data\_write[2] = data & 0xFF;

m\_i2c.write(m\_addr, data\_write, 3);

}

void VL6180x::VL6180x\_setRegister16bit(uint16\_t registerAddr, uint16\_t data)

{

char data\_write[4];

data\_write[0] = (registerAddr >> 8) & 0xFF;

data\_write[1] = registerAddr & 0xFF;

data\_write[2] = (data >> 8) & 0xFF;

data\_write[3] = data & 0xFF;

m\_i2c.write(m\_addr, data\_write, 4);

}

void printIdentification(struct VL6180xIdentification \*temp){

pc.printf("Model ID = ");

pc.printf("%d\r\n",temp->idModel);

pc.printf("Model Rev = ");

pc.printf("%d",temp->idModelRevMajor);

pc.printf(".");

pc.printf("%d\r\n",temp->idModelRevMinor);

pc.printf("Module Rev = ");

pc.printf("%d",temp->idModuleRevMajor);

pc.printf(".");

pc.printf("%d\r\n",temp->idModuleRevMinor);

pc.printf("Manufacture Date = ");

pc.printf("%d",((temp->idDate >> 3) & 0x001F));

pc.printf("/");

pc.printf("%d",((temp->idDate >> 8) & 0x000F));

pc.printf("/1");

pc.printf("%d\r\n",((temp->idDate >> 12) & 0x000F));

pc.printf(" Phase: ");

pc.printf("%d\r\n",(temp->idDate & 0x0007));

pc.printf("Manufacture Time (s)= ");

pc.printf("%d\r\n",(temp->idTime \* 2));

pc.printf("\r\n\r\n");

}

void released(){

if(debounceTimer.read\_ms() > 50){

but = 1;

}

}

void pressed(){

debounceTimer.reset();

debounceTimer.start();

}

void Button(){

myButton.rise(&pressed);

myButton.fall(&released);

}

void action0(){

pc.printf("X0=%d X1=%d X2=%d X3=%d X4=%d

X5=%d X6=%d\r\n",x0,x1,x2,x3,x4,x5,x6);

Button();

pc.printf("Wait to Start\r\n\r\n");

if (but == 1){

pc.printf("Start\r\n\r\n");

but = 0;

x0 = 0;

x1 = 1;

}

wait\_ms(500);

}

void action1(){

pc.printf("X0=%d X1=%d X2=%d X3=%d X4=%d

X5=%d X6=%d\r\n",x0,x1,x2,x3,x4,x5,x6);

wait\_ms(100); // delay .1s

pc.printf("Get identification\r\n\r\n");

sensor.getIdentification(&identification);

printIdentification(&identification);

x1 = 0;

x2 = 1;

wait\_ms(500);

}

void action2(){

int i = sensor.VL6180xInit();

pc.printf("X0=%d X1=%d X2=%d X3=%d X4=%d

X5=%d X6=%d\r\n",x0,x1,x2,x3,x4,x5,x6);

if(i != 0){

pc.printf("FAILED TO INITALIZE\r\n\r\n");

x2 = 0;

x0 = 1;

}

else{

pc.printf("SUCCESS TO INITIALIZE\r\n\r\n");

x2 = 0;

x3 = 1;

}

wait\_ms(500);

}

void action3(){

pc.printf("X0=%d X1=%d X2=%d X3=%d X4=%d

X5=%d X6=%d\r\n",x0,x1,x2,x3,x4,x5,x6);

pc.printf("Set default\r\n\r\n\r\n");

sensor.VL6180xDefautSettings();

x3 = 0;

x4 = 1;

wait\_ms(500);

}

void action4(){

pc.printf("X0=%d X1=%d X2=%d X3=%d X4=%d

X5=%d X6=%d\r\n",x0,x1,x2,x3,x4,x5,x6);

pc.printf("Ambient Light Level (Lux) = ");

pc.printf("%f\r\n\r\n\r\n",sensor.getAmbientLight(GAIN\_1) );

x4 = 0;

x5 = 1;

wait\_ms(500);

}

void action5(){

pc.printf("X0=%d X1=%d X2=%d X3=%d X4=%d

X5=%d X6=%d\r\n",x0,x1,x2,x3,x4,x5,x6);

pc.printf("Distance measured (mm) = ");

pc.printf("%d\r\n\r\n\r\n", sensor.getDistance() );

x5 = 0;

x6 = 1;

wait\_ms(500); // delay 1s

}

void action6(){

pc.printf("X0=%d X1=%d X2=%d X3=%d X4=%d

X5=%d X6=%d\r\n\r\n\r\n",x0,x1,x2,x3,x4,x5,x6);

Button();

if (but == 1){

pc.printf("Restart!\r\n\r\n");

but = 0;

x6 = 0;

x0 = 1;

wait\_ms(500); // delay 1s

}

else {

x6 = 0;

x4 = 1;

}

}

void grafcet(){

while (1){

if (x0 == 1) action0();

if (x1 == 1) action1();

if (x2 == 1) action2();

if (x3 == 1) action3();

if (x4 == 1) action4();

if (x5 == 1) action5();

if (x6 == 1) action6();

}

}

int main() {

x0 = 1; x1 = 0; x2 = 0; x3 = 0; x4 = 0; x5 = 0; x6 = 0; but = 0;

grafcet();

}