# 14 LETTING GO of the MBED LIBRARIES – Code Snips

Rev. 19.11.2016

**Note: These Code Snips are taken straight from the book chapter; i.e. the “Program Examples”. In some cases therefore they are not complete programs.**

/\*Program Example 14.1: Sets up a digital output pin using control registers, and flashes an led.

\*/

// function prototypes

void delay(void);

//Define addresses of digital i/o control registers, as pointers to volatile data

#define FIO2DIR0 (\*(volatile unsigned char \*)(0x2009C040))

#define FIO2PIN0 (\*(volatile unsigned char \*)(0x2009C054))

int main() {

FIO2DIR0=0xFF; // set port 2, lowest byte to output

while(1) {

FIO2PIN0 |= 0x01; // OR bit 0 with 1 to set pin high

delay();

FIO2PIN0 &= ~0x01; // AND bit 0 with 0 to set pin low

delay();

}

}

//delay function

void delay(void){

int j; //loop variable j

for (j=0;j<1000000;j++) {

j++;

j--; //waste time

}

}

Program Example 14.1: Manipulating control registers to flash an LED

for (i=1;i<=3;i++){

FIO0PIN0 |= 0x02; // set port 2 pin 1 high (mbed pin 25)

delay();

FIO0PIN0 &= ~0x02; // set port 2 pin 1 low

delay();

}

Program Example 14.2 (code fragment): Controlling a second LED output

/\* Program Example 14.3: Uses digital input and output using control registers, and flashes an LED. LEDS connect to mbed pins 25 and 26. Switch input to pin 9.

\*/

// function prototypes

void delay(void);

//Define Digital I/O registers

#define FIO0DIR0 (\*(volatile unsigned char \*)(0x2009C000))

#define FIO0PIN0 (\*(volatile unsigned char \*)(0x2009C014))

#define FIO2DIR0 (\*(volatile unsigned char \*)(0x2009C040))

#define FIO2PIN0 (\*(volatile unsigned char \*)(0x2009C054))

//some variables

char a;

char b;

char i;

int main() {

FIO0DIR0=0x00; // set all bits of port 0 byte 0 to input

FIO2DIR0=0xFF; // set port 2 byte 0 to output

while(1) {

if ((FIO0PIN0&0x01)==1){ // bit test port 0 pin 0 (mbed pin 9)

//note correction here. Extra brackets inserted

//as operator == has higher precedence than &

a=0x01; // this reverses the order of LED flashing

b=0x02; // based on the switch position

}

else {

a=0x02;

b=0x01;

}

FIO2PIN0 |= a;

delay();

FIO2PIN0 &= ~a;

delay();

for (i=1;i<=3;i++){

FIO2PIN0 |= b;

delay();

FIO2PIN0 &= ~b;

delay();

}

} //end while loop

}

//delay function

void delay(void){

int j; //loop variable j

for (j=0;j<1000000;j++) {

j++;

j--; //waste time

}

}

Program Example 14.3: Combined digital input and output

/\* Program Example 14.4: Sawtooth waveform on DAC output. View on oscilloscope. Port 0.26 is used for DAC output, i.e. mbed Pin 18

\*/

// function prototype

void delay(void);

// variable declarations

int dac\_value; //the value to be output

//define addresses of control registers, as pointers to volatile data

#define DACR (\*(volatile unsigned long \*)(0x4008C000))

#define PINSEL1 (\*(volatile unsigned long \*)(0x4002C004))

int main(){

PINSEL1=0x00200000; //set bits 21-20 to 10 to enable analog out on P0.26

while(1){

for (dac\_value=0;dac\_value<1023;dac\_value=dac\_value+1){

DACR=(dac\_value<<6);

delay();

}

}

}

//delay function

void delay(void){

int j; //loop variable j

for (j=0;j<1000000;j++) {

j++;

j--; //waste time

}

}

Program Example 14.4: Saw tooth output on the DAC

/\* Program Example 14.5: A bar graph meter for ADC input, using control registers to set up ADC and digital I/O

\*/

// variable declarations

char ADC\_channel=1; // ADC channel 1

int ADCdata; //this will hold the result of the conversion

int DigOutData=0; //a buffer for the output display pattern

// function prototype

void delay(void);

//define addresses of control registers, as pointers to volatile data

//(i.e. the memory contents)

#define PINSEL1 (\*(volatile unsigned long \*)(0x4002C004))

#define PCONP (\*(volatile unsigned long \*)(0x400FC0C4))

#define AD0CR (\*(volatile unsigned long \*)(0x40034000))

#define AD0GDR (\*(volatile unsigned long \*)(0x40034004))

#define FIO2DIR0 (\*(volatile unsigned char \*)( 0x2009C040))

#define FIO2PIN0 (\*(volatile unsigned char \*)( 0x2009C054))

int main() {

FIO2DIR0=0xFF;// set lower byte of Port 2 to output, this drives bar graph

//initialise the ADC

PINSEL1=0x00010000; //set bits 17-16 to 01 to enable AD0.1 (mbed pin 16)

PCONP |= (1 << 12); // enable ADC clock

AD0CR = (1 << ADC\_channel) // select channel 1

| (4 << 8) // Divide incoming clock by (4+1), giving 4.8MHz

| (0 << 16) // BURST = 0, conversions under software control

| (1 << 21) // PDN = 1, enables power

| (1 << 24); // START = 1, start A/D conversion now

while(1) { // infinite loop

AD0CR = AD0CR | 0x01000000; //start conversion by setting bit 24 to 1,

//by ORing

// wait for it to finish by polling the ADC DONE bit

while ((AD0GDR & 0x80000000) == 0) { //test DONE bit, wait till it’s 1

}

ADCdata = AD0GDR; // get the data from AD0GDR

AD0CR &= 0xF8FFFFFF; //stop ADC by setting START bits to zero

// Shift data 4 bits to right justify, and 2 more to give 10-bit ADC

// value - this gives convenient range of just over one thousand.

ADCdata=(ADCdata>>6)&0x03FF; //and mask

DigOutData=0x00; //clear the output buffer

//display the data

if (ADCdata>200)

DigOutData=(DigOutData|0x01); //set the lsb by ORing with 1

if (ADCdata>400)

DigOutData=(DigOutData|0x02); //set the next lsb by ORing with 1

if (ADCdata>600)

DigOutData=(DigOutData|0x04);

if (ADCdata>800)

DigOutData=(DigOutData|0x08);

if (ADCdata>1000)

DigOutData=(DigOutData|0x10);

FIO2PIN0 = DigOutData; // set port 2 to Digoutdata

delay(); // pause

}

}

//delay function

void delay(void){

int j; //loop variable j

for (j=0;j<1000000;j++) {

j++;

j--; //waste time

}

}

Program Example 14.5: Applying the ADC as a bar graph

/\*Program to explore ADC conversion times, programming

control registers directly. ADC value is transferred to DAC,

while an output pin is strobed to indicate conversion duration.

Observe on oscilloscope

tjw 14.5.11\*/

/\*mbed pin allocation

analog input: pin 16

analog (ie DAC) output: pin 18, ie port 0.26

led/adc duration indication: pin 26 \*/

//#include "mbed.h"

// variable declarations

char ADC\_channel=1; // ADC channel 1

int ADCdata; //this will hold the result of the conversion

//define addresses of control registers in use, as pointers to volatile data

//(i.e. the memory contents)

#define PINSEL1 (\*(volatile unsigned long \*)(0x4002C004))

#define PCONP (\*(volatile unsigned long \*)(0x400FC0C4))

#define AD0CR (\*(volatile unsigned long \*)(0x40034000))

#define AD0GDR (\*(volatile unsigned long \*)(0x40034004))

#define DACR (\*(volatile unsigned long \*)(0x4008C000))

#define FIO2DIR0 (\*(volatile unsigned char \*)( 0x2009C040))

#define FIO2PIN0 (\*(volatile unsigned char \*)( 0x2009C054))

//lowest 8 bits of Port 2

// function prototype

void delay(void);

int main() {

FIO2DIR0=0xFF; // set lower bits port 2 to output

PINSEL1=0x00210000; //set bits 21-20 to 10 to enable analog output

//(mbed pin 18)

//and bits 17-16 to 01 to enable AD0.1 (mbed pin 16)

//initialise the ADC. Values for different sections of AD0CR are shifted and ORed in

PCONP |= (1 << 12); // enable ADC clock

AD0CR = (1 << ADC\_channel) // channel 1

| (4 << 8) // Divide incoming clock by (4+1), giving 4.8MHz

| (0 << 16) // BURST = 0, software control

| (0 << 17) // CLKS = 0 (11-bits)

| (1 << 21) // PDN = 1

| (0 << 22) // TEST1:0 = 00

| (1 << 24) // START = 1 start A/D conversion

| (0 << 27); // EDGE = 0

while(1){

// start A/D conversion by modifying bits in the AD0CR register

AD0CR &= (AD0CR & 0xFFFFFF00);

FIO2PIN0 |= 0x01; // OR bit 0 with 1 to set pin high

AD0CR |= (1 << ADC\_channel) | (1 << 24);

// wait for it to finish by polling the ADC DONE bit

while((AD0GDR & 0x80000000) == 0) {

}

FIO2PIN0 &= ~0x01; // AND bit 0 with 0 to set pin low

ADCdata = AD0GDR; // get the data from AD0GDR

AD0CR &= 0xF8FFFFFF; //stop ADC by setting START bits to zero

// shift data 4 bits to right justify, and 2 more to give 10-bit ADC value

ADCdata=(ADCdata>>6)&0x03FF; //and mask

DACR=(ADCdata<<6);

//could be merged with previous line, but separated for clarity

//delay(); //insert delay if wished

}

}

void delay(void){ //delay function.

int j; //loop variable j

for (j=0;j<1000000;j++) {

j++;

j--; //waste time

}

}

Program Example 14.6: Applying ADC, DAC and digital output, to measure conversion duration (full listing)