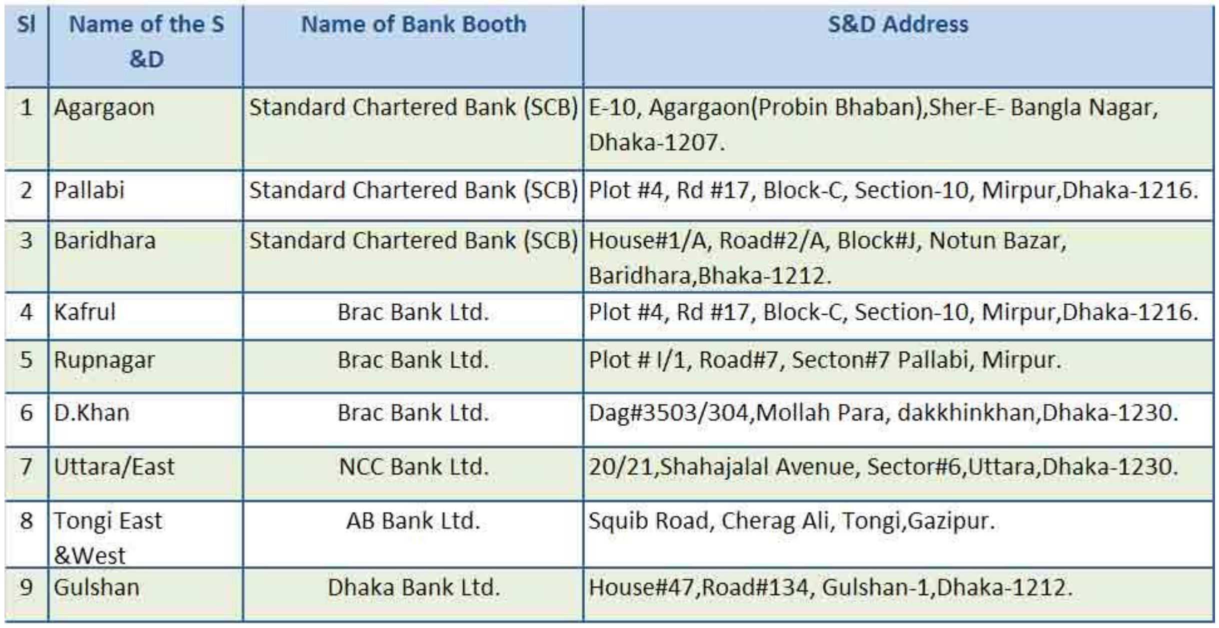
APPENDIX A



Sl	Name of the S &D	Name of Bank Booth	S&D Address
1	Agargaon	Standard Chartered Bank (SCB)	E-10, Agargaon(Probin Bhaban),Sher-E- Bangla Nagar, Dhaka-1207.
2	Pallabi	Standard Chartered Bank (SCB)	Plot #4, Rd #17, Block-C, Section-10, Mirpur,Dhaka-1216.
3	Baridhara	Standard Chartered Bank (SCB)	House#1/A, Road#2/A, Block#J, Notun Bazar, Baridhara,Bhaka-1212.
4	Kafrul	Brac Bank Ltd.	Plot #4, Rd #17, Block-C, Section-10, Mirpur,Dhaka-1216.
5	Rupnagar	Brac Bank Ltd.	Plot # I/1, Road#7, Secton#7 Pallabi, Mirpur.
6	D.Khan	Brac Bank Ltd.	Dag#3503/304,Mollah Para, dakkhinkhan,Dhaka-1230.
7	Uttara/East	NCC Bank Ltd.	20/21,Shahajalal Avenue, Sector#6,Uttara,Dhaka-1230.
8	Tongi East &West	AB Bank Ltd.	Squib Road, Cherag Ali, Tongi,Gazipur.
9	Gulshan	Dhaka Bank Ltd.	House#47,Road#134, Gulshan-1,Dhaka-1212.

Figure A.1- DESCO service booths

APPENDIX B

```
/*! \file gsm.c \brief GSM function library. */

#define F_CPU 16000000UL


#include <avr/io.h>

#include <util/delay.h>

#include <avr/interrupt.h>

#include <string.h>

#include "lcd_lib.h"

#include "gsm.h"

//////////////////////////////////////////////////// Possible GSM Replies ////////////////////////////////////////////////


const unsigned char  CR_LF[]          = "\r\n";                                  // Carriage Return
Line Feed

const unsigned char     OK[]          = "OK\r\n";                                // OK string

const unsigned char  CREG[]           = "+CREG: 0,1\r\n";               // Home network
registered

const unsigned char     CALL_RDY[]          = "Call Ready\r\n";                  // Modem is
ready

const unsigned char  RING[]           = "RING\r\n";                     // Incoming call


const unsigned char  CMTI[]     = "+CMTI:";              // New Message arrived

const unsigned char  READY[]          = "> ";                                    // Phone ready to
receive text message

const unsigned char     PWR_DWN[]= "NORMAL POWER DOWN\r\n";// Power down the
modem
```

```
//////////////////////////////////////////////// Possible strings /////////////////////////////////////////////

const unsigned char  INVALID[]  = "Invalid";              // Invalid msg received

const unsigned char  LIGHT_ON[] = "Light on";             // "Light on" msg from user

const unsigned char  LIGHT_OFF[]= "Light off";            // "Light off" msg from user

const unsigned char  STATUS[]   = "Status";               // Status of light

const unsigned char  PhnNo[]= "+8801554306021";              // Any phn no.

// Initialize pointer

const unsigned char  *searchStrings[7]  = {CR_LF, OK, CREG, CALL_RDY, CMTI,
READY, PWR_DWN};

////////////////////////////////////////// AT-Command set used /////////////////////////////////////////////


const unsigned char  AT[]                = "AT\r";

const unsigned char  ATH[]               = "ATH\r";
          // Hangup call

const unsigned char  ATE0[]     = "ATE0\r";                        // Echo off

const unsigned char  AT_CREG[]     = "AT+CREG?\r";
          // Check Network

const unsigned char  AT_CMGF[]  = "AT+CMGF=1\r";                   // Text Mode

const unsigned char  AT_CNMI[]  = "AT+CNMI=2,1,0,0,0\r";           // Identification of new
SMS

const unsigned char  AT_CPMS[]  = "AT+CPMS=\"SM\",\"SM\",\"SM\"\r"; // Preferred
storage

const unsigned char  AT_CMGS[]     = "AT+CMGS=\"";
          // Send message

const unsigned char  AT_CMGD[]    = "AT+CMGD=1\r";                                          //
Delete message at index 1
```

```
const unsigned char  AT_CMGR[]     = "AT+CMGR=1\r";                                        //
Read from index 1

//////////////////////////////////////////////// Debug variables ////////////////////////////////////////////////

/* USART initialization

 This function set correct baudrate and functionality of the USART.

 */


void usart_init( unsigned int baudrate, short dspeed )

{

          UBRRH = (unsigned char) (baudrate>>8);//Setting baudrate

   UBRRL = (unsigned char) baudrate;     //Setting baudrate


   if (dspeed == NormalSpeed)

   {

                   UCSRA &= ~( 1 << U2X );

   }

          else

          {

                   UCSRA |= ( 1 << U2X );                              //Double the speed

          }


   UCSRB |= ( 1 << RXEN ) | ( 1 << TXEN ); //Enable receiver and transmitter

   UCSRC |= ( 1 << URSEL ) | ( 1 << UCSZ1 ) | ( 1 << UCSZ0 );  //8 data bit
```

```
        //////////// Reset Variable ///////////
        sec_count = 0;    // increased at timer0 interrupt
        RingIndex = 0;//////////////////////////////////////

        usart_rx_reset();        // Reset receive buffer
        usart_rx_on();                          // Enable Rx interrupt
        sei();
}
/* RX interrupt disable + reset all variable
 */
void usart_rx_reset( void )
{
   UCSRB &= ~(1<<RXCIE);         // Disable RX interrupt
   rx_i = rx_wr_i = 0;           // Reset variables
   rx_overflow = rx_ack = 0;
   rx_buffer[ rx_wr_i ] = '\0';
}
/*  RX interrupt disable This function disable usart's Rx interrupt
 */
void usart_rx_off( void )
{
        UCSRB &= ~(1<<RXCIE);        // Disable RX interrupt
}
/*  RX interrupt enable This function enable usart's Rx interrupt
 */
```

```
void usart_rx_on( void )
{
    UCSRB |= ( 1 << RXCIE );
}

/* Initialize the modem This function start and config the modem
 */
int modem_init( unsigned short *port, unsigned short pin )
{
    LCDGotoString(0,1, "Init start..");
            Delay_s(1);
            SetSearchString( CALLRDY_ );
            usart_rx_on();

            *port |= (1<<pin);
            Delay_s(3);
            *port &= ~(1<<pin);

            if (check_acknowledge(35) > 0)
            {
                        SetSearchString( CREG_ );
                        Delay_s(1);

                        LCDGotoString(0,1, "Check Network..");
```

```
Delay_s(1);


check_network:
usart_putStr(AT_CREG);
usart_rx_on();


if (check_acknowledge(7) > 0)
{
        SetSearchString( OK_ );              //Set OK to be search string
        usart_putStr( ATE0 );                //Send turn echo off
        usart_rx_on();                       //Receiver on


        if( check_acknowledge(5) > 0 )       //Echo off = OK
        {
                usart_putStr(AT_CMGF);           //Send Text Mode
                usart_rx_on();                   //Receiver on


        if( check_acknowledge(5) > 0 )
                {
                usart_putStr(AT_CPMS);       //Send preferred storage
                usart_rx_on();
if( check_acknowledge(8) > 0 ) //Preferred storage = OK
                {
                        usart_putStr(AT_CNMI);     //Send preferred indication of new
messages
                        usart_rx_on();


                        if( check_acknowledge(8) > 0 )   //Preferred indication = OK
                                {
                                return 1;
                                }
                                else
                        {
                                // CNMI failed
                                }
                }
                else
                {
                        // CPMS failed
                }
                }
                else
                {
                // CMGF failed
                }
                }
```

```
                        else
                        {
                                // ATE0 failed
                        }
                }
                else
                {
                        goto check_network;
                }

        }
        else
        {
                LCDGotoString(0,1, "Init Problem");
                Delay_s(5);
        }
}
/*
 *      Set desired search string
 */

void SetSearchString( unsigned char Response )
{
   usart_rx_off();                                      // Disable RX interrupt
        searchFor = searchStrings[Response];  //Set desired search string
```

```
    searchStr = Response;                //Used in rx_ISR

    rx_i = 0;

}

/*

 *  Print string through USART

 */


void usart_putStr( const unsigned char *str )

{

    for( ;*str != '\0'; )

    {

        usart_putchar( *str++ );

    }

}


/*

 *        Put char in transmit buffer

 */


short usart_putchar( unsigned char data )

{


    unsigned int i;



    for(i=0; !(UCSRA & (1<<UDRE)); i++)// Wait for empty transmit buffer
    {
        if( i > TX_WAIT )              // How long one should wait
        {
            return -1;                 // Transmission fail, Give feedback to function caller
        }
    }
    UDR = data;                        // Start transmition
}
int check_acknowledge( unsigned short timeOut)
{
            sec_count = 0;

            //Wait loop
            for( ;( rx_ack == 0 ) && ( sec_count < timeOut ); )
            {

                        /////// for debug ////////
                        char str[3];
                        itoa(sec_count,str,10);
                        LCDGotoString(14,1,str);
                        /////////////////////////

                        _delay_ms(100);
            }
        if( rx_ack > 0 )        //Everything worked out fine
    {
        rx_ack = 0;         //Reset flag and return 1
        return 1;
    }
    else                    //A timeout could result from no acknowledge, wrong acknowledge or
buffer overrun
    {
        usart_rx_reset();  //Reset buffer and interrupt routine
        return 0;          //Timed out, or wrong acknowledge from phone
    }

}


/* Send message         Return Value:
     1 Success, message sent
     -1 No "> " from phone
   -2 No message sent acknowledge
 */


int send_msg( unsigned char* str, unsigned char* no )
{
        SetSearchString( READY_ );      //Set READY to be search string
        usart_putStr( AT_CMGS );        //Send AT command
        usart_putStr(no);                                //Send Phn no
        usart_putStr("\"\r");                    //Send CR
    usart_rx_on();                //Receiver on

        if( check_acknowledge(6) > 0 )        // Get ">"
        {
                SetSearchString( OK_ );
                usart_putStr(str);                        //Send msg
                usart_putchar( 26 );
                usart_rx_on();

                if( check_acknowledge(30) > 0 ) //Send SMS successfully
                {
                        return 1;
                }
                else
                {
                        // Msg sending failed
                }
        }
        else
        {
                // No ">"
        }
```

```
}

/*  Check modem

This function 1st check if the modem is hang by sending AT command. If so then it restart the
modem.

 */

void checkError( void )

{

        short errorCount=0;


        AT:

        usart_rx_reset();

        SetSearchString( OK_ );

        usart_putStr(AT);

        usart_rx_on();


        if ( check_acknowledge(8) > 0 )

        {

                LCDGotoString(0,1, "AT OK");

                Delay_s(1);

        }


        else

        {

                sec_count = 0;

                errorCount++;

                if (errorCount > 3)    // Modem is hang. Restart the modem.
}


                {
                        errorCount = 0;

                        LCDGotoString(0,1, "Initiating shutdown..");
                        Delay_s(2);

                        SetSearchString( PWRDWN_);
                        usart_rx_on();

                        PORTB |= (1<<PB6);          // PortB1 is connected with GSM_PWR
pin of modem

                        if (check_acknowledge(12) > 0)
                        {
                                PORTB &= ~(1<<PB1);
                                modem_init(&PORTB,6);
                        }
                }
                else
                {
                        while(sec_count < 8);
                        goto AT;
                }
        }
}
```

```
/*  Delay function

One second delay function.  Parameter takes the value of second.

 */

void Delay_s(char sec)

{

        char i,j;

        for(j=0; j<sec; j++)

        {

                for(i=0; i<10; i++)

                {

                        _delay_ms(100);

                }

        }

}
```

```
//Code for Sending SMS

#include <mega8.h>

#include <delay.h>

#include <stdio.h>

#include <stdlib.h>

#include <delay.h>

// LCD module connections

sbit LCD_RS at PORTD2_bit;

sbit LCD_EN at PORTD3_bit;

sbit LCD_D4 at PORTD4_bit;

sbit LCD_D5 at PORTD5_bit;

sbit LCD_D6 at PORTD6_bit;

sbit LCD_D7 at PORTD7_bit;


sbit LCD_RS_Direction at DDD2_bit;

sbit LCD_EN_Direction at DDD3_bit;

sbit LCD_D4_Direction at DDD4_bit;

sbit LCD_D5_Direction at DDD5_bit;

sbit LCD_D6_Direction at DDD6_bit;

sbit LCD_D7_Direction at DDD7_bit;

// End LCD module connections

unsigned char si=0,qi=0;

char q[160];


interrupt [USART_TXC] void usart_transmit_isr(void)
```

```
{if(qi!=si)
UDR=q[si++];
}
void sendmsg(char *s)
{
qi=0;
si=1;
while(*s)
{q[qi++]=*s++;
}
UDR=q[0];
}
void main(void)
{

  Lcd_Init();                       // Initialize LCD
  Lcd_Cmd(_LCD_CLEAR);              // Clear display
  Lcd_Cmd(_LCD_CURSOR_OFF);
UCSRA=0x00;
UCSRB=0x48;
UBRRH=0x00;
UBRRL=0x33;
char a[50];
int i;
for(i=0;i<50;i++)
{
a[i]=0x01+i;
}
#asm ("sei")
sendmsg("AT\r\n");
delay_ms(2000);
sendmsg("AT+CMGF=1\r\n");
delay_ms(2000);
sendmsg("AT+CSMP=17,167,0,241\r\n");
delay_ms(2000);
sendmsg("AT+CMGS=\"+8801520090283\"\r\n");
delay_ms(2000);
sendmsg(a);
delay_ms(2000);
UDR=0x1A;
}
```