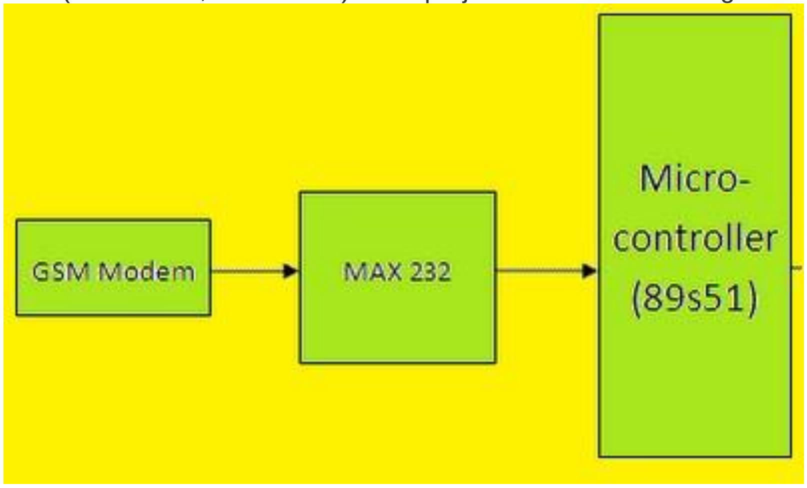
GSM modem interfacing with microcontroller 8051 for SMS control of industrial equipments

This is a beginner tutorial in which a GSM modem is being interfaced with the microcontroller AT89s51 for SMS communication. The SMS can be send and received for the data sharing and situation information and control.

We can use it as a remote control of industrial machines or we can sue it for home automation or we can use it for the security of home or offices. The sending SMS through GSM modem when interfaced with microcontroller or PC is much simpler as compared with sending SMS through Modem in PDU Mode .Text message may be sent through the modem by interfacing only three signals of the serial interface of modem with microcontroller i.e.,TxD,RxD and GND.
In this scheme RTS and CTS signals of serial port interface of GSM Modem are connected with each other.
The transmit signal of serial port of microcontroller is connected with transmit signal (TxD) of the serial interface of GSM Modem while receive signal of microcontroller serial port is connected with receive signal (RxD) of serial interface of GSM Modem.

The COMPIM Serial Port Model shown in the schematic diagram developed in Proteus VSM is equivalent to the serial interface of GSM Modem. Sending SMS Messages from a Computer / PC Using AT Commands (AT+CMGS, AT+CMSS) 8051 projects based on message received on mobile



The following are the AT Commands and sequence of events performed for sending text message to a mobile phone through GSM Modem interfaced with microcontroller :

1. First select the text mode for SMS by sending the following AT Command to GSM Modem : AT+CMGF = 1 . This command configures the GSM modem in text mode.

2. Send the following AT Command for sending SMS message in text mode along with mobile number to the GSM Modem : AT+CMGS =+923005281046 . This command sends the mobile number of the recipient mobile to the GSM modem.

3. Send the text message string (“hello!”) to the GSM Modem This is a test message from UART”

4. Send ASCII code for CTRL+Z i.e., 0x1A to GSM Modem to transmit the message to mobile phone. After message string has been sent to the modem, send CTRL+Z to the micro-controller, which is equivalent to 0x1A (ASCII value) Every AT command is followed by i.e. carriage return and line feed you are giving line feed first and carriage return after that.
“\r” stands for carriage return

The SMS message in text mode can contain only 140 characters at the most. It depends upon the amount of information collected from GPS Engine that you need at the base station
for tracking vehicle or person
The most important string of GPS is “$GPRMC…” which contains the minimum information required in tracking a target.
Additionally you may need the information regarding the number of satellites that are visible to the GPS receiver. So it depends upon how much information you need to pack in 140 characters of SMS in text mode.

Code for the interfacing the GSM modem with microcontroller 8051 for just testing of connection and serial communication is under:-

```
#include<at89x51.h> // include at89x51 . h
#include<stdio.h>// include stdio . h
#include<stdlib.h>// include stdlib . h
void initialize_GSM_modem(void);
void initialize_serialcommunication(void);
unsigned char Command_CMGF[]="AT+CMGF=1\r";
// AT+CMGF for selecting Text Mode
unsigned char CtrlZ=0x1A;
// CTRL+Z for sedning SMS after the message has been entered
unsigned char Command_CMGS[]="AT+CMGS =+9233385xxxxx\r";
// recepient mobile number
unsigned char Command_AT[]="AT\r";
unsigned char msg02[]="Hello!";
void delay(void){
unsigned int i;
for(i=0;i<50;i++);
}
void delay2(void){
unsigned int i;
for(i=0;i<25000;i++);
}
void main (void) {
initialize_GSM_modem();
initialize_serialcommunication();
while (1) {
;
}
}
void initialize_GSM_modem(void){
delay2();
puts(Command_AT);
delay2();
puts(Command_CMGF);
delay2();
puts(Command_CMGS);
delay2();
puts(msg02);
delay2();
while(!TI); TI = 0;SBUF = 0x1A;
}
void initialize_serialcommunication(void){
TMOD = 0x20;
SCON = 0x50;
TH1  = 0xFD;
TL1  = 0xFD;
TR1  = 1;
TI = 1;
}
```