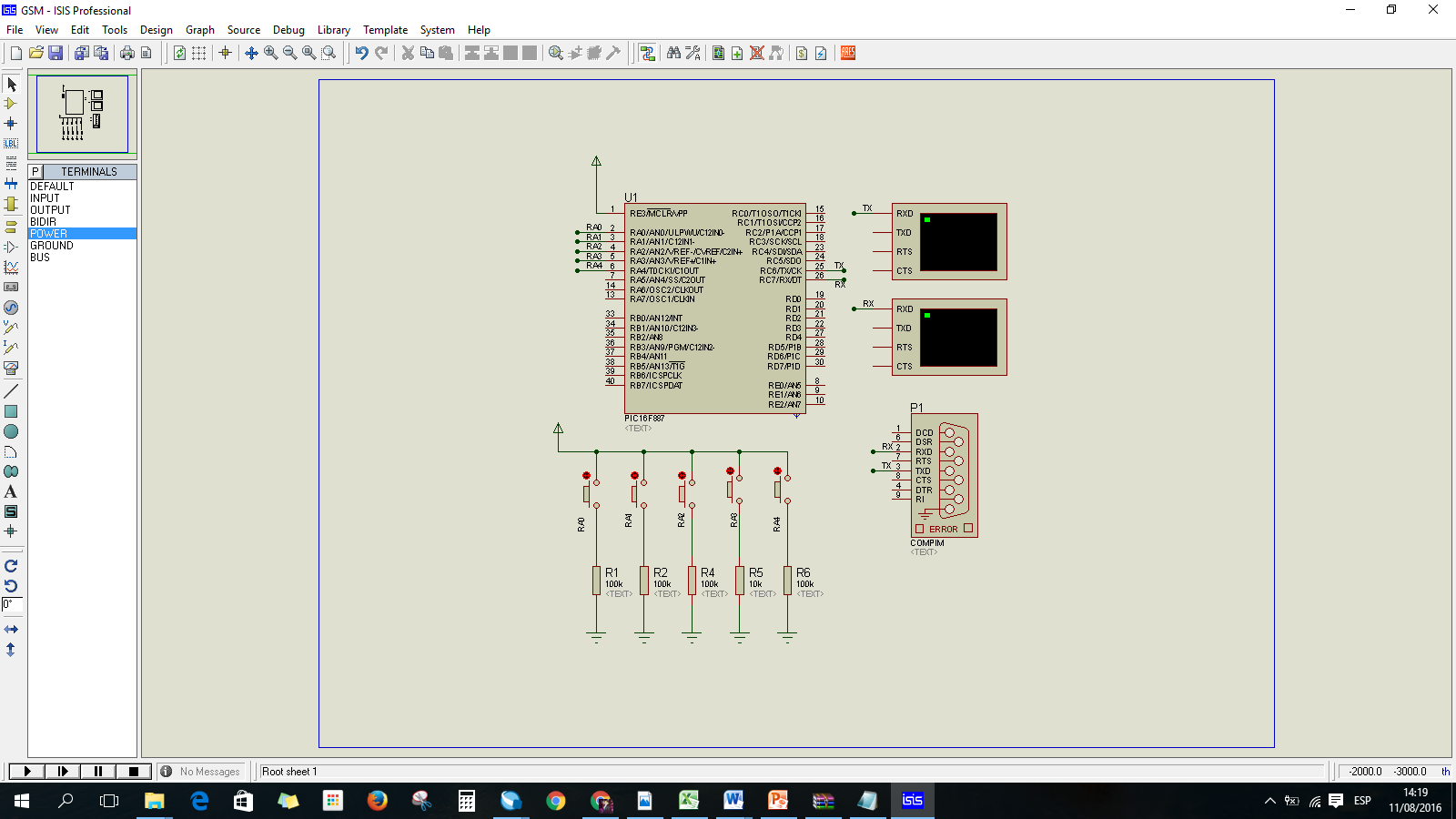
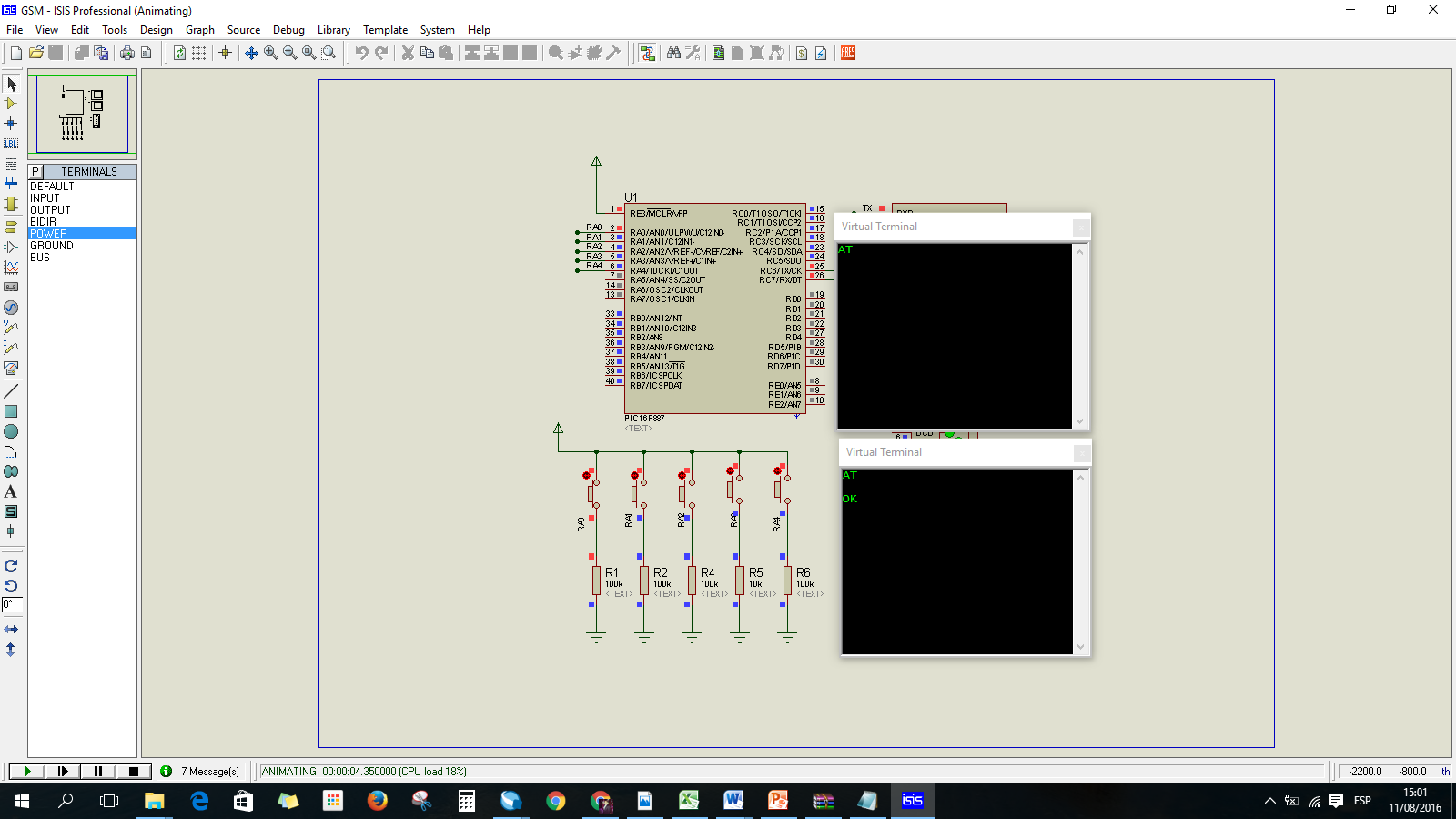
**TUTORIAL GSM SIM900**

**CIRCUITO:**

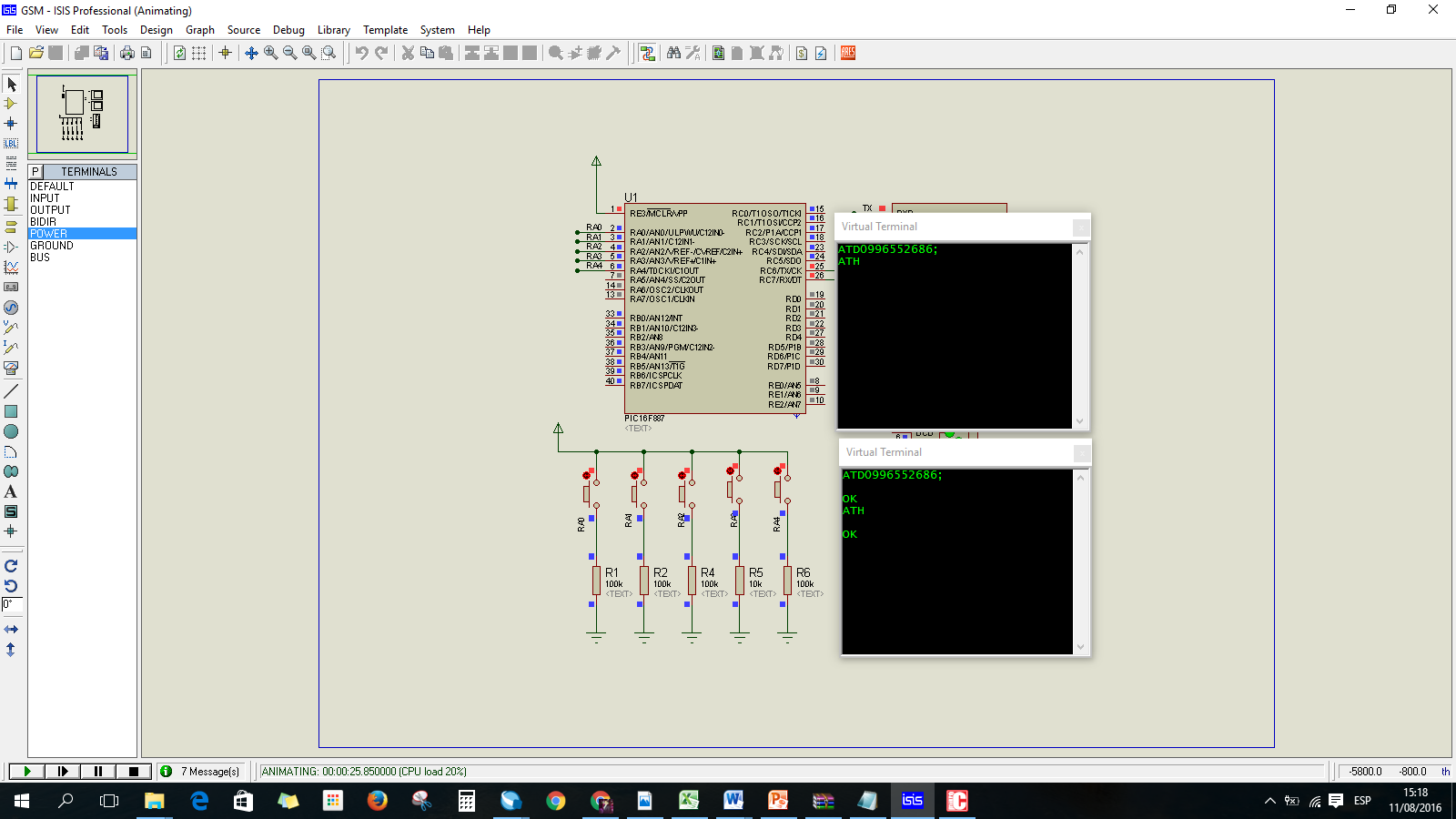


El COMPIM fue utilizado para comunicarse con el módulo GSM SIM900 a través del integrado MAX232.

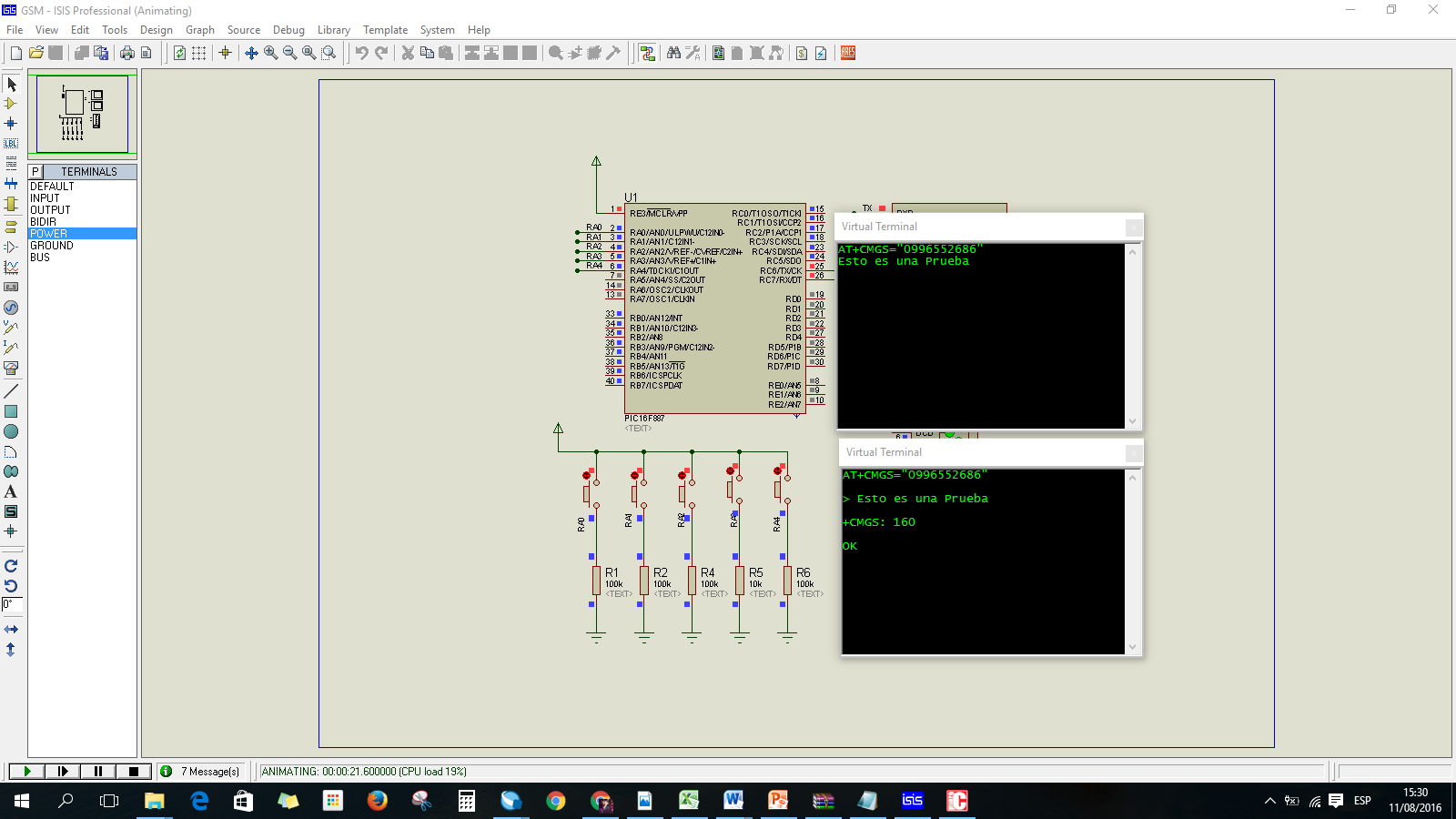
Cuando se inicializa el circuito, se lo prueba ingresando y soltando la botonera RA0 que enviará el comando: “AT”, que sirve para conocer si el módulo SIM900 está conectado y operando. Si el módulo está conectado recibirá como respuesta “OK”.



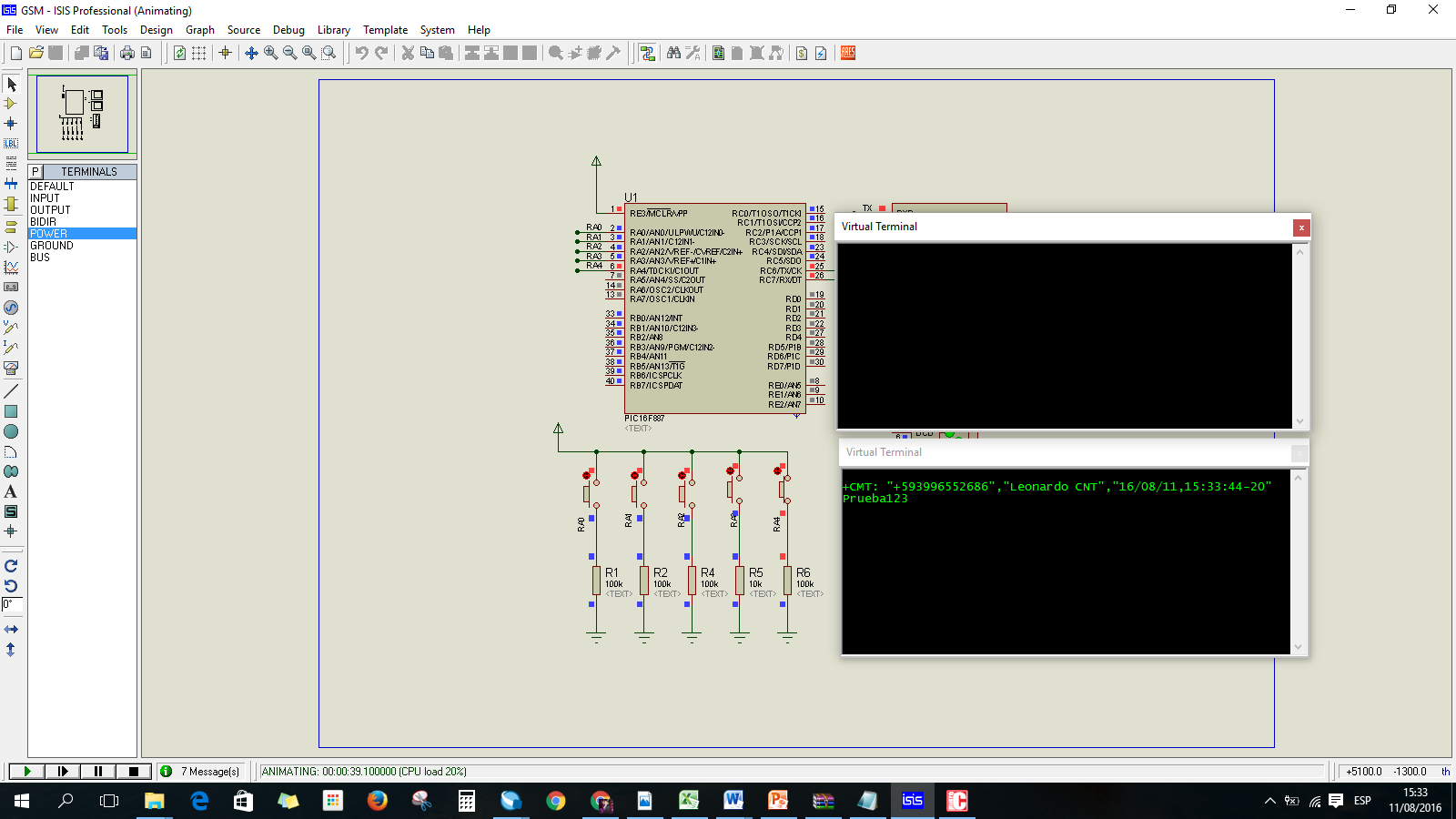
Presionando y soltando la botonera RA1 enviará el comando ATD que sirve para realizar una llamada, seguido del número a donde se va a realizar la llamada. Finaliza con el retorno de carro. Si el comando está bien, el módulo enviará como respuesta “OK”. Después 15 segundos enviará el comando ATH, que sirve para colgar la llamada que se está realizando, y el módulo responderá con “OK”.



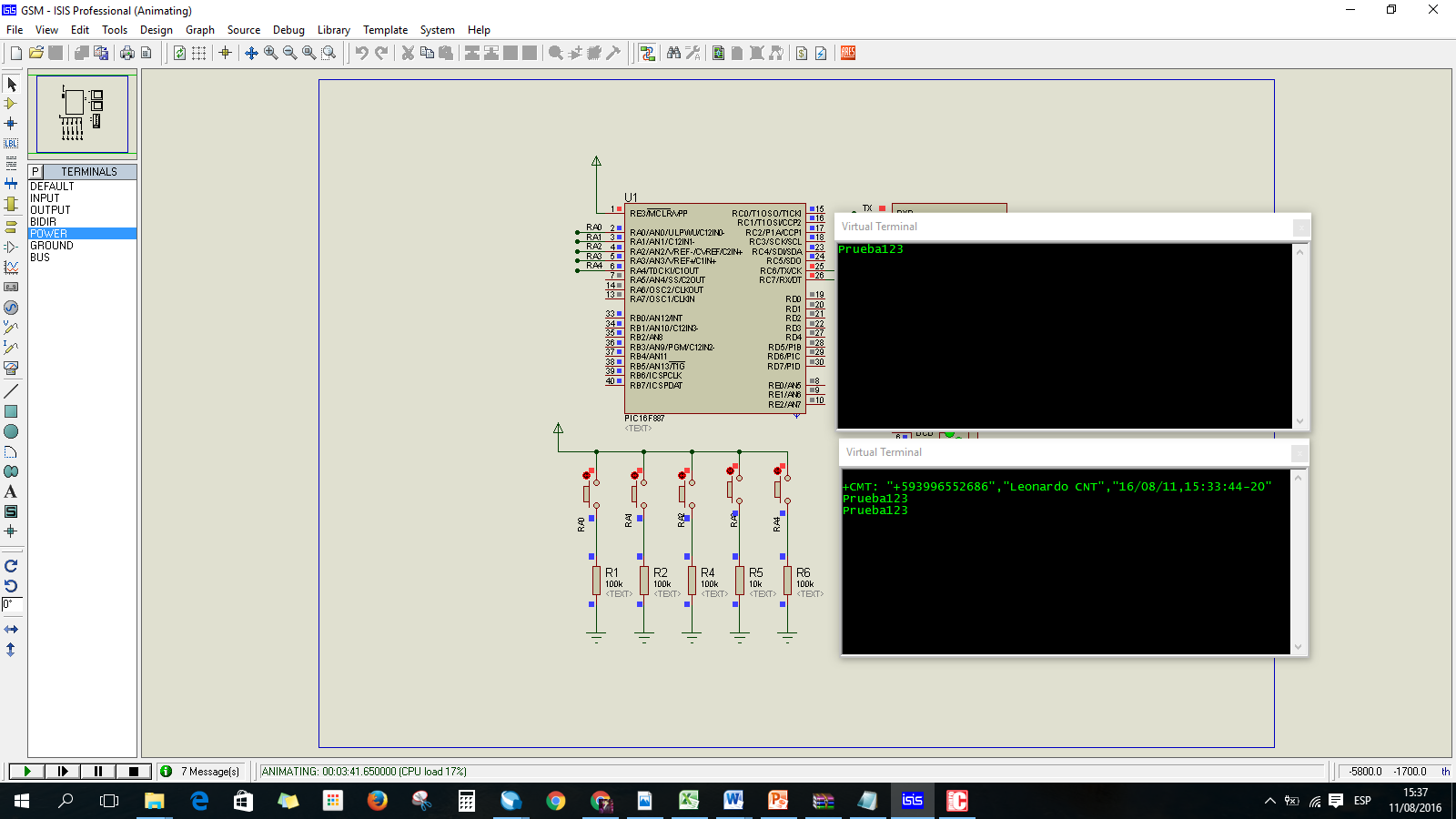
Presionando y soltando la botonera RA2 enviará el comando AT+CMGS que sirve para un mensaje, seguido del número a donde se va a enviar el mensaje. Finaliza con el retorno de carro. Se escribirá el mensaje y finalizará el envío mandando la tecla Control+Z. Si el comando está bien, el módulo enviará como respuesta “OK”.



Para recibir un mensaje se debe primero dejar presionada la tecla RA4, y una vez que se reciban datos, se la puede soltar.



Como observamos en la figura anterior, el módulo muestra transmitirá la trama recibida que incluye el número del cual se envió el mensaje, junto con el nombre correspondiente a ese número en el caso de que se encuentre guardado en la tarjeta SIM, la fecha y la hora que se recibió el mensaje, y al último se encuentra el contenido del mensaje recibido que es la parte más importante a ser tratada. Aplastando y soltando la botonera RA3, se transmitirá solamente el contenido del mensaje de la trama recibida.



**ANEXO:**

**CODIGO:**

**/\***

**\* Nombre del Proyecto:**

**GSM.c**

**\* Nombre del Autor:**

**Ing. Leonardo Muñoz.**

**\* Description:**

**Explicación de comandos AT con el SIM900**

**\* Test configuration:**

**MCU: PIC16F887**

**Oscillator: HS, 08.0000 MHz**

**SW: mikroC PRO for PIC**

**\* NOTES:**

**\*/**

**/\*Header\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/**

**unsigned short i;**

**void init(){**

**ANSEL = 0; // Configure AN pins as digital**

**ANSELH = 0;**

**UART1\_Init(9600); // Initialize USART module**

**// (8 bit, 9600 baud rate, no parity bit...)**

**PORTA = 255;**

**TRISA = 255; //Utiliza PORTA como entrada**

**TRISB=0; //Utiliza PORTB como salida**

**PORTB=0;**

**TRISC=0x80; //Utiliza RC7 como entrada**

**PORTC=0x80;**

**}**

**void transmision(char \*cadena){ //Funcion que permite enviar cadenas de caracteres de cualquier tamaño vía UART**

**int size,i;**

**size=strlen(cadena);**

**Delay\_ms(1000);**

**for(i=0; i<size; i++){**

**UART1\_Write(cadena[i]);**

**Delay\_ms(10);**

**}**

**}**

**void main() {**

**char buffer1[75]; //Cadena de caracteres para guardar un valor recibido vía UART**

**char \*p; //Puntero para recorrer un arreglo**

**char ctrlz=26; //Caracter ascii de la tecla Control+Z**

**int size=0;int i=0;**

**int flag=0; int cont=0;**

**init();**

**memset(buffer1,'\0',strlen(buffer1)); // Borra el contenido de buffer1**

**while (1) {**

**if(RA0\_bit==1){ //Es verdadero si se aplasta la botonera RA0**

**transmision("AT\r"); //Transmite el comando AT vía UART y finaliza con el retorno de carro, es decir, la tecla enter**

**}**

**if(RA1\_bit==1){ //Es verdadero si se aplasta la botonera RA1**

**transmision("ATD0996552686;\r"); //Transmite el Comando ATD que sirve para realizar una llamada, seguido del número a donde se va a realizar la llamada. Finaliza con el retorno de carro.**

**Delay\_ms(15000); //Espera 15 segundos hasta que el celular destino recibe la llamada.**

**transmision("ATH\r"); //Transmite el comando ATH que sirve para terminar la llamada, seguido del retorno de carro**

**}**

**if(RA2\_bit==1){ //Es verdadero si se aplasta la botonera RA2**

**transmision("AT+CMGS=\"0996552686\"\r"); //Transmite el comando AT+CMGS que sirve para enviar un mensaje de texto, seguido del número donde se enviará el mensaje. Finalizando con el retorno de carro se pedirá que escriba el mensaje que va a enviar**

**transmision("Esto es una Prueba\r\n"); //Este es el contenido que tendrá el mensaje**

**UART1\_Write(ctrlz); //Para poder enviar el mensaje se tiene que ingresar la tecla ascii de Control+Z**

**}**

**//La sentencia "IF(RA3\_bit==1)" de abajo se debe de utilizar si he recibido algún dato a través del puerto RX vía UART**

**if(RA3\_bit==1){ //Es verdadero si se aplasta la botonera RA3**

**Delay\_ms(1000); //Espero 1 segundo**

**cont=0; //Inicializo la variable**

**p=buffer1; //El puntero p apunta a la primera dirección de memoria de la variable buffer1, es decir, p=buffer1[0]**

**while(\*p!='\0'){ //El puntero recorrerá la variable buffer1 hasta que apunte al fin de la cadena**

**if(flag==1){ //Entro en esta sentencia solamente para escribir el contenido del mensaje**

**UART1\_Write(\*p); //Transmitiré el mensaje vía UART**

**Delay\_ms(100); //Espera 100 milisegundos**

**}**

**else {flag=0;}**

**if(\*p=='P'){ //Entro en la sentencia si el puntero encuentra el caracter 'P', que de manera predeterminada será cuando comience el contenido del mensaje**

**flag=1; //Esta bandera "flag" indicará si encontré o no el caracter P en la transmision del mensaje**

**UART1\_Write(\*p); //Transmitirá el caracter 'P' encontrado en el mensaje**

**Delay\_ms(100); //Espera 100 milisegundos**

**}**

**p++;**

**}**

**memset(buffer1,'\0',strlen(buffer1)); //Borra el contenido de la variable buffer1 utilizada para guardar datos a través de RX**

**flag=0; //Encero la bandera**

**}**

**//La sentencia "IF(RA4\_bit==1)" de abajo se debe de utilizar en el caso de que vaya a recibir algún dato a través del puerto RX vía UART**

**if(RA4\_bit==1){ //Es verdadero si se aplasta la botonera RA4**

**while(UART1\_Data\_Ready()==1){ // Es verdadero mientras un dato sea recibido a través del puerto RX vía UART**

**buffer1[cont]=UART1\_Read(); //Voy guardando cada caracter recibido en la variable buffer1**

**cont++; //Este contador incrementará para ir guardando los valores en la variable buffer1**

**}**

**}**

**}**

**}**