Generic solution to control the WIFIBOT with an external PC :

You need to create 2 threads, one for sending commands and the other for receiving.
For that you need to create a TCP/IP socket connecting to the WIFIBOT IP with port 15000.

So to communicate with the WIFIBOT you need just to receive “buf2” and to send “buf” :

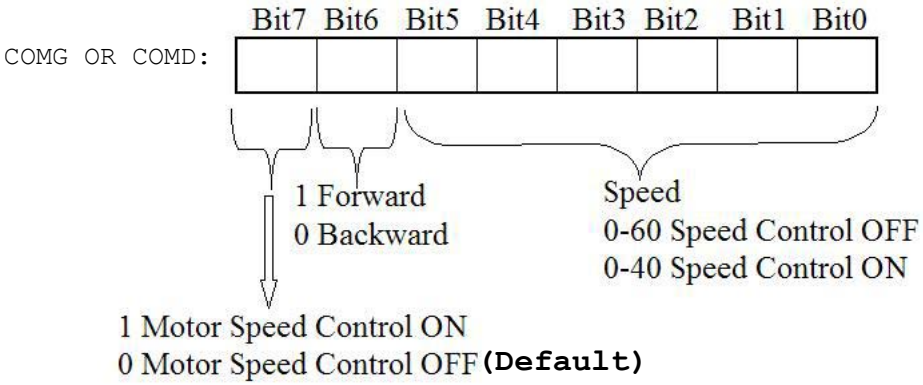
```
unsigned char buf[2], buf2[7];
```

There is the details of the communication :

Sending commands :

```
buf[0]=(unsigned char)(comg);//left motor board command
buf[1]=(unsigned char)(comd);//right motor board command
```

“comg” and “comd” is generated depending of the way we want the robot to move :



Receiving sensors values :

buf2[0]	battery value (0-255)
buf2[1]	front left wheels ticks (0-40 ticks/41ms)
buf2[2]	rear left wheels ticks (0-40 ticks/41ms)
buf2[3]	front right wheels ticks (0-40 ticks/41ms)
buf2[4]	rear right wheels ticks (0-40 ticks/41ms)
buf2[5]	Left infrared sensor (0-150 cm)
buf2[6]	Right infrared sensor (0-150 cm)

Example using C++ and MFC sockets :
Class : “MyClass” :

void InitSocket(void);

void Close(void);

Thread_Send();

Thread_Receive();

/*
UINT Thread_Send (LPVOID p)

UINT Thread_Receive (LPVOID p)
*/

```
CAsyncSocket so;//MFC socket

bool running, running2 ; //to end the Thread

Cstring ip;robot ip

int port;//robot port

unsigned char buf[2], buf2[7]; //send receive buffer

unsigned char comg, comd; //motor board command


void MyClass::InitSocket(void)
{
   so.Create();
   int status = so.Connect(ip,porttemp);
   running=true;
   running2=true;
   AfxBeginThread(Thread_Receive,this);
   AfxBeginThread(Thread_Send,this);
}

void MyClass::Close(void)
{
      running2=false;
      running=false;
      so.Close();
}
```

```
void MyClass::Thread_Send(void)
{
	while(running)
	{
		buf[0]=(unsigned char)(comg);//left motor board command
		buf[1]=(unsigned char)(comd);//right motor board command
		so.Send(&buf, 2, 0);
		Sleep(60); //16 Hz
	}
}

UINT MyClass::Thread_Send (LPVOID p)
{
   MyClass *me = (MyClass *)p;
   me-> Thread_Send ();
   return 0;
}

void MyClass::Thread_Receive(void)
{
	while(running2)
	{
		int rcvnbr=so.Receive(&buf2,7,0);
		if (rcvnbr==7)
		{
		//Do what you want with the buffer
		}
	}
}

UINT MyClass::Thread_Receive (LPVOID p)
{
   MyClass *me = (MyClass *)p;
   me-> Thread_Receive ();
   return 0;
}
```