Note on Communication protocol

Da te : 27/08/2013 – RRU/NMS/Thiyagu

In Online_v2 development we are using generic structure based communication protocol to communicate command & response between Online_v2 program and MCM rabbit card. Origin of this communication protocol is from 15 Meter NCRA CMS system. Where it's working without any problem for last one year. For our online_v2 development, we have trimmed and modified the structure to suit our GMRT sub-system requirement.

Connection is made using TCP/IP networking socket programming between Online_v2 and MCM program. Connection remains persistent means connections are not broken after exchanging information. Connections are broken only if MCM device is power off.

Command Structure :

```
typedef struct
{
  int seq;                          // Unique Sequence
  char timestamp[64];               //  Timestamp of command
  char system_name[16];             // System Name for which command is
  char op_name[16];                 // Operation to perform ( Init/Set/Mon/Reset )
  short int number_param;           // Number of parameter
  char parameter_name[32][16];      // Parameter Name
  char Argument_Ch1[32][16];        // Channel One argument
  char Argument_Ch2[32][16];        // Channel Two argument
 } cmd;
```

Response structure :

```
typedef struct
{
 int response_type;                 // Response type
 int seq;                           // Sequence number
 char timestamp[64];                // Time stamp
 char system_name[16];              // System name
 char Mon_raw[64][8];               // 64 channel raw data
```

```
 char Mon_sum[32][64];                  // Monitoring summary prepared from 64 channel raw data
 short int num_resp_msg;                // Number of Response Message
 char response_message[32][64];         // Response message from MCM
} resp;
```

```
typedef struct
{
   int sub_num;
   int num_in_sub;
   char ant_name[32][8];
 }sub;

typedef struct
{
   int usernb;
   sub s_a[4];
   char timestamp[26];
} user;

typedef struct
{
   user Tx[4];

} setdata;

typedef struct
{
  char antenna_name[10];
  cmd  CMD;
} ANT_CMD;
```

Structure for Servo Communication :

```
typedef struct
{
  int seq;
  char timestamp[64];
  char system_name[16]; // Not required in servo communication
 // char cmd_code[4];
  char op_name[16];
  short int number_param;
  char para_name[32][16];
  char para_value[32][16];
 // // Not required in servo communication
 } servocmd;
```

```
typedef struct
{
 int seq;
 char timestamp[64];
 char system_name[16];
 int response_code;     // immediate =1 ( ACK or NAK ) event=3 & final =2  data mon = 4
 int response_type;     // success =1 or failure = 2
 char response_msg[50];  //Accepted ,not accepted, sysntax error, irrelvant
                                    command,incomplete,unknown + Event
 short int num_resp_msg;
 char para_name[32][32];
 char para_value[32][32];
} servoresp;


typedef struct
{
  char antenna_name[10];
  servocmd  SRVCMD;
} SRVANT_CMD;
```

Size of command structure is 1638 bytes and size of response structure is 4698 bytes.

We have been using this generic communication protocol structure in C03 antenna testing & in two antenna 4 sub-system each Lab test set-up. We have not found any major problem related to communication. This generic structure based communication protocol is working very well for our Online_v2 need.

In term of reducing time interval, we are continuously working on it. With the help of powerful machines & optimization we will be able to reduce the time interval as well.

Comparison between Present Online, Teleset-Abccom & Online_v2 protocol :

Present Online do all the hex code generation and decoding in Online machine and pass it on to communication handler through PC router on a serial link @9.6 kbps. All the intelligence is in Online code.

Teleset-Abccom protocol differ in a sense that most of the intelligence has been shifted to Abccom software. Teleset only gives higher level ASCII command like English text for ex: if setnew set1.dat . Abccom will decode what it means by if setnew set1.dat, generate the corresponding hex code for that and send that to MCM 10 on serial link & get the response and form the response packet from the raw data sent by MCM 10 which can be sent to Teleset.

In Online_v2, all the bit pattern generation and decoding of response from raw data is done by MCM card so we just have to send higher level command from Online_v2 by an agreed upon communication protocol. We are following a structure based communication protocol where we are sending a command structure to a particular MCM card and we are getting a response structure.
In Online_v2 most of the intelligence is shifted in the MCM card itself.