

```

1      const PORT = 34012;
2      const PORT_TX = 34001;
3      const ADDRESS = '192.168.0.134';
4      const IHC_STATE_FOLDER = '0_userdata.0.IHC.';
5      const IHC_CONNECTION_ACTIVE = IHC_STATE_FOLDER + 'Info.connectionActive';
6      const dgram = require('dgram');
7      const server = dgram.createSocket('udp4');
8      server.on('error', (err) => {
9          console.error(`server error:\n${err.stack}`);
10         server.close();
11     });
12     server.on('message', (msg, rinfo) => {
13         const resArray = msg.toString().split("\r");
14         if(resArray.length > 1 ){
15             resArray.pop();
16             //console.log('server got:')
17             resArray.forEach(WriteToStates);
18             setState(IHC_CONNECTION_ACTIVE,{val:true,expire: 300});
19         }
20     });
21     async function WriteToStates(value) {
22         const sig_name_arr = value.split("\n");
23         const state_name = sig_name_arr[0] + '_' + sig_name_arr[1] + '_' +
24             sig_name_arr[2] + '_' + sig_name_arr[3];
25         const sig_name = IHC_STATE_FOLDER + '0.' + state_name;
26         var common = {};
27         var value0;
28         common['name'] = state_name;
29         common['ack'] = true;
30         switch(sig_name_arr[2]){
31             case '0': // digital Type
32                 common['type'] = 'boolean';
33                 if(sig_name_arr[4]!==0) value0 = true;
34                 else value0 = false;
35                 break;
36
37             case '1': // analog Type
38                 common['type'] = 'number';
39                 value0 = parseInt(sig_name_arr[4]);
40                 break;
41
42             case '2': // String Type
43                 common['type'] = 'string';
44                 value0 = sig_name_arr[4].replaceAll("&deg;", "°");
45                 break;
46
47             case '3': // Temperature Type
48                 common['type'] = 'string';
49                 common['unit'] = '°C';
50                 value0 = sig_name_arr[4].replaceAll("&deg;C", "");
51                 break;
52
53             case '4': // Time Type
54                 common['type'] = 'string';
55                 value0 = sig_name_arr[4];
56                 break;
57
58             case '5': // Date Type
59                 common['type'] = 'string';
60                 value0 = sig_name_arr[4];
61                 break;
62             default:
63                 value0=0;
64         }
65         if( existsState(sig_name) ) {
66             await setStateAsync(sig_name, value0, true);
67         } else {
68             common.name = state_name;
69             await createStateAsync(sig_name, value0, common);
70         }
71         //console.log(sig_name + ": " + value0);

```

```

71     }
72     server.on('listening', () => {
73         const address = server.address();
74         console.log(`server listening ${address.address}:${address.port}`);
75     });
76     server.bind(PORT+1);
77     createState(IHC_CONNECTION_ACTIVE , false,true, {type: 'boolean'});
78     //import { Buffer } from 'node:buffer';
79     const message = Buffer.from('0');
80     const client = dgram.createSocket('udp4');
81     var sch_WDT = schedule('*/*30 * * * *', async function () {await
82     client.send(message, PORT, ADDRESS);});
83     const client_tx = dgram.createSocket('udp4');
84     //client_tx.send(message_tx, PORT_TX, ADDRESS);
85     async function StatesChange(obj) {
86         const sig_name_arr = obj.id.substring(obj.id.lastIndexOf('.') + 1).split("_");
87         //console.log(sig_name_arr.length + ' --Val ' + obj.state.val + ' --ACK ' +
88         obj.state.ack + ' --From ' + obj.state.from);
89         if(sig_name_arr.length == 4 ){
90             // 0: Module ID 1: Type(0:IN,1:OUT,2:Var) 2:DataType 3: Signal Number
91             if(sig_name_arr[0]<256 && sig_name_arr[1]==0 && sig_name_arr[2]<256 &&
92             sig_name_arr[3]<256){
93                 if(sig_name_arr[2]==0){//DataTpye digital
94                 const message_tx = Buffer.from([4, 0,0,0, // Num Of Data (4bytes)
95                 2, 0,0,0, // Digital signal
96                 sig_name_arr[0],0,0,0, // Module ID
97                 sig_name_arr[3], 0,0,0, // Signal Number
98                 (obj.state.val==false)?2:1, 0,0,0, // operation (0=toggle, 1=Set 2 = Reset)
99                 0, 0,0,0, 0, 0,0,0, 0, 0,0,0, 0, 0,0,0, 0, 0,0,0, 0, 0,0,0]); // zero
100                 fill
101                 client_tx.send(message_tx, PORT_TX, ADDRESS);
102                 //console.log(sig_name_arr[0] + '-' + sig_name_arr[3] + ' --Val ' +
103                 obj.state.val + ' --ACK ' + obj.state.ack + ' --From ' +
104                 obj.state.from);//.state.val);
105             }
106         }
107     }
108     };
109     //on({id:[].concat([IHC_STATE_FOLDER , '0.5_0_0_15']), change: "ne"},
110     StatesChange);
111     //on({id:'0_userdata.0.IHC.0.5_0_0_15'/*5 0 0 15*/, change: "ne"}, StatesChange);
112     on({id:/^0_userdata\.0\.IHC\.0\.d+_0_[0-1]_d+$/, change: "ne", ack:false},
113     StatesChange);
114     // console.log('List Subscriptions:');
115     // console.log(getSubscriptions());
116     // bei Scriptstop
117     onStop (function scriptStop () {
118         clearSchedule(sch_WDT);
119         server.close();
120         client.close();
121         client_tx.close();
122     }, 1000);

```