```c
/*Compilação*/
//      gcc servidor_tcp.c play_alarm.c -o servidor_tcp

/*Execução*/
// ./servidor_tcp

//-------------------------------------------------------------------------------------------------------
/*Includes do programa*/
#include <stdio.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <stdlib.h>
#include <string.h>
#include <unistd.h>
#include <pthread.h>
#include <termios.h>
#include <fcntl.h>
#include <time.h>
#include <signal.h>
#include "play_alarm.h"

//-------------------------------------------------------------------------------------------------------
/*Defines*/
#define DEVICE_FILE             "/dev/ttyAMA0"
#define DEBUG


//-------------------------------------------------------------------------------------------------------
/*Variáveis globais*/
char alarm_setor[] = {0, 0};
pid_t alarm_pid;
int uart0_filestream = -1;
unsigned short porta_4000 = 4034;
unsigned short porta_8080 = 8034;


//-------------------------------------------------------------------------------------------------------
/*Função de Log do sistema*/

void Log_data(char *clienteAddr, int setor, int dispositivo, int estado)
{
        FILE *fd;                                                       //Descritor do arquivo

    time_t tempo;                                               //Struct da biblioteca time.h
        struct tm *data;
        time(&tempo);
        data=localtime(&tempo);

        fd = fopen("Client_log.txt", "a+");     //Abre o arquivo para incluir

        //HH:MM:SS - DD/MM/AAAA - Cliente <IP>; Setor <1-6>; Dispositivo <X>; Estado <0-1>
        fprintf(fd, "%d:%d:%d - %d/%d/%d",(*data).tm_hour,(*data).tm_min,(*data).tm_sec,
(*data).tm_mday,(*data).tm_mon+1,(*data).tm_year+1900);
        fprintf(fd, " - Cliente %s; Setor %d; Dispositivo %d; Estado %d\n", clienteAddr, setor,
dispositivo, estado);

        fclose(fd);                                                     //Fecha o arquivo
}


//-------------------------------------------------------------------------------------------------------
/* Comunicação UART*/

//Configura os parametros da comunicacao UART
void Uart_Config(int uart0_filestream)
{
        struct termios options;
```

```
66          tcgetattr(uart0_filestream, &options);
67          options.c_cflag = B9600 | CS8 | CLOCAL | CREAD; // Set baud rate
68          options.c_iflag = IGNPAR;
69          options.c_oflag = 0;
70          options.c_lflag = 0;
71          tcflush(uart0_filestream, TCIFLUSH);
72          tcsetattr(uart0_filestream, TCSANOW, &options);
73  }
74
75  //Recebimento de float via UART
76  void Recebe_float (int uart0_filestream, float *retorno)
77  {
78          float temperatura;
79          uart0_filestream = open(DEVICE_FILE, O_RDWR | O_NOCTTY |O_NDELAY);
80
81          int rx_length = read(uart0_filestream, &temperatura, sizeof(float));
82                  if (rx_length < 0)
83                          perror("Falha na leitura");
84
85                  else if (rx_length == 0)
86                          printf("Nenhum dado disponivel\n");
87
88                  else
89                          printf("\n|%i bytes lidos|", rx_length);
90
91
92          close(uart0_filestream);
93
94  #ifdef DEBUG
95
96          printf("\n\nRetorno float: %f\n\n\n", temperatura);
97          *retorno = temperatura;
98
99  #endif
100 }
101
102 //Recebimento de char via UART
103 void Recebe_char (char * retorno, int tamanho, int uart0_filestream)
104 {
105         int i;
106
107         uart0_filestream = open(DEVICE_FILE, O_RDWR | O_NOCTTY |O_NDELAY);
108         int rx_length = read(uart0_filestream, retorno, tamanho);
109
110         if (rx_length < 0)
111                 perror("Falha na leitura");
112
113         else if (rx_length == 0)
114                 printf("Nenhum dado disponivel\n");
115
116         else
117                 printf("| %i bytes lidos |", rx_length);
118
119         close(uart0_filestream);
120
121 }
122
123 //Envio de char via UART
124 void Envia_char (char *Comando, int tamanho, int uart0_filestream)
125 {
126
127 #ifdef DEBUG
128         printf("Comando : %X %d %X\n", Comando[0], Comando[1], Comando[2]);
129 #endif
130
131         uart0_filestream = open(DEVICE_FILE, O_RDWR | O_NOCTTY |O_NDELAY);
132         int tx_length = write(uart0_filestream, Comando, tamanho);
```

```
133                 if (tx_length != tamanho)
134                         printf("Erro na transmissao - UART TX\n");
135
136         close(uart0_filestream);
137 }
138
139 //-------------------------------------------------------------------------------------------------------
140 /*Thead de verificação dos sensores de presença, portas e janelas*/
141
142 void *Verifica_sensor()
143 {
144         int i;
145         char estado_sensores[9];
146         char TODOS_OS_SENSORES = 0xA4;
147
148         while(1)
149         {
150                 Envia_char (&TODOS_OS_SENSORES, 1, uart0_filestream);
151                 sleep(2);
152                 Recebe_char (estado_sensores, 9, uart0_filestream);
153
154 #ifdef DEBUG
155                 /*for (i = 0; i < 9; i++)
156                 {
157                         printf("S[%d] %d, ",i, estado_sensores[i] );
158                 }
159                 printf("\n");*/
160 #endif
161
162                 /*Monitoramento do Setor 1*/
163                 if(alarm_setor[0] == 1)
164                 {
165                         for (i = 0; i < 3; i++)
166                         {
167                                 if(estado_sensores[i] != 0)
168                                 {
169                                         //Toca Alarme
170                                         printf("Toca o ALARME !!!\n");
171                                         play_alarm_music(&alarm_setor[0]);
172                                 }
173                                 else
174                                         sleep(2);
175
176                         }
177                 }
178
179                 /*Monitoramento do Setor 2*/
180                 if(alarm_setor[1] == 1)
181                 {
182                         for (i = 3; i < 9; i++)
183                         {
184                                 if(estado_sensores[i] != 0)
185                                 {
186                                         //Toca Alarme
187                                         printf("Toca o ALARME !!!\n");
188                                         play_alarm_music(&alarm_setor[1]);
189                                 }
190                                 else
191                                         sleep(2);
192
193                         }
194                 }
195         }
196 }
197
198
199 //-------------------------------------------------------------------------------------------------------
```

```
200  /*Funções de envio de dados ao cliente*/
201
202  void Envia_float_cliente(int socketCliente, struct sockaddr_in clienteAddr, float temperatura) {
203
204          if(send(socketCliente, &temperatura, sizeof(float), 0) != sizeof(float))
205                  printf("Erro no envio - send()\n");
206
207  }
208
209  void Envia_char_cliente(int socketCliente, struct sockaddr_in clienteAddr, char mensagem) {
210
211          if(send(socketCliente, &mensagem, sizeof(char), 0) != sizeof(char))
212                          printf("Erro no envio - send()\n");
213
214  }
215
216
217  //-------------------------------------------------------------------------------------------------
218  /*Tratamento da informação recebida do cliente*/
219
220
221  void enviaArduino(char setor, char dispositivo, char estado, int socketCliente, struct sockaddr_in
     clienteAddr)
222  {
223          //Declaração das variaveis
224          char SETOR_TEMPERATURA  = 0xA1;
225          char SETOR_PRESENCA             = 0xA2;
226          char SETOR_PORTAS               = 0xA3;
227          char SETOR_AC                   = 0xB1;
228          char SETOR_LAMPADAS             = 0xB2;
229          char SETOR_ALARME               = 0xB3;
230          char comando[3];
231          char resposta;
232          float temperatura;
233
234          //Divisão dos dispositivos em setores
235          switch(setor)
236          {
237                  //Menu temperatura
238                  case 1:
239                          //Passando parâmetros para comando
240                          comando[0] = SETOR_TEMPERATURA;
241                          comando[1] = dispositivo;
242                          comando[2] = estado;
243
244                          //Enviando o comando para o arduino
245                          Envia_char (comando, 2, uart0_filestream);
246
247                          //Recebe resposta
248                          Recebe_float(uart0_filestream, &temperatura);
249
250                          //Envia resposta ao cliente
251                          Envia_float_cliente(socketCliente, clienteAddr, temperatura);
252
253  #ifdef DEBUG
254                          switch(dispositivo)
255                          {
256                                  case 1:
257                                          printf("Temperatura - Externa\n");
258                                          break;
259
260                                  case 2:
261                                          printf("Temperatura - Interna da Sala\n");
262                                          break;
263
264                                  case 3:
265                                          printf("Temperatura - Interna do quarto\n");
```

```
266                                         break;
267                         }
268 #endif
269                         break;
270
271                 //Menu Presenca
272                 case 2:
273
274                         comando[0] = SETOR_PRESENCA;
275                         comando[1] = dispositivo;
276                         comando[2] = estado;
277                         Envia_char (comando, 2, uart0_filestream);
278                         Recebe_char(&resposta, 1, uart0_filestream);
279                         Envia_float_cliente(socketCliente, clienteAddr, resposta);
280
281 #ifdef DEBUG
282                         switch(dispositivo)
283                         {
284                                 case 1:
285                                         printf("Presença - Entrada Principal\n");
286                                         break;
287
288                                 case 2:
289                                         printf("Presença - Entrada de Serviço\n");
290                                         break;
291
292                                 case 3:
293                                         printf("Presença - Garagem\n");
294                                         break;
295
296                         }
297 #endif
298                         break;
299
300                 //Menu Portas/Janelas
301                 case 3:
302
303                         comando[0] = SETOR_PORTAS;
304                         comando[1] = dispositivo;
305                         comando[2] = estado;
306                         Envia_char (comando, 2, uart0_filestream);
307                         Recebe_char(&resposta, 1, uart0_filestream);
308                         Envia_float_cliente(socketCliente, clienteAddr, resposta);
309
310 #ifdef DEBUG
311                         switch(dispositivo)
312                         {
313
314                                 case 1:
315                                         printf("Estado - Porta principal\n");
316                                         break;
317
318                                 case 2:
319                                         printf("Estado - Porta de serviço\n");
320                                         break;
321
322                                 case 3:
323                                         printf("Estado - Porta garagem\n");
324                                         break;
325
326                                 case 4:
327                                         printf("Estado - Janela da sala\n");
328                                         break;
329
330                                 case 5:
331                                         printf("Estado - Janela do quarto\n");
332                                         break;
```

```
333
334                                 case 6:
335                                         printf("Estado - Janela da cozinha\n");
336                                         break;
337                         }
338     #endif
339                         break;
340
341                 //Menu Ar-condicionado
342                 case 4:
343
344                         comando[0] = SETOR_AC;
345                         comando[1] = dispositivo;
346                         comando[2] = estado;
347                         Envia_char (comando, 3, uart0_filestream);
348
349     #ifdef DEBUG
350                         switch(dispositivo)
351                         {
352                                 case 1:
353                                         printf("AC - Sala - %d\n", estado);
354                                         break;
355
356                                 case 2:
357                                         printf("AC - Quarto - %d\n", estado);
358                                         break;
359                         }
360     #endif
361                         break;
362
363                 //Menu Lampadas
364                 case 5:
365
366                         comando[0] = SETOR_LAMPADAS;
367                         comando[1] = dispositivo;
368                         comando[2] = estado;
369                         Envia_char (comando, 3, uart0_filestream);
370
371     #ifdef DEBUG
372                         switch(dispositivo)
373                         {
374                                 case 1:
375                                         printf("Lampada - Entrada principal - %d\n", estado);
376                                         break;
377
378                                 case 2:
379                                         printf("Lampada - Entrada de serviço - %d\n", estado);
380                                         break;
381
382                                 case 3:
383                                         printf("Lampada - Garagem - %d\n", estado);
384                                         break;
385
386                                 case 4:
387                                         printf("Lampada - Sala - %d\n", estado);
388                                         break;
389
390                                 case 5:
391                                         printf("Lampada - Quarto - %d\n", estado);
392                                         break;
393
394                                 case 6:
395                                         printf("Lampada - Cozinha - %d\n", estado);
396                                         break;
397                         }
398     #endif
399                         break;
```

```
400
401                 //Menu Alarme
402                 case 6:
403
404                         comando[0] = SETOR_ALARME;
405                         comando[1] = dispositivo;
406                         comando[2] = estado;
407                         Envia_char (comando, 3, uart0_filestream);
408
409                         if(dispositivo == 1)
410                                 alarm_setor[0] = estado;
411
412                         if(dispositivo == 2)
413                                 alarm_setor[1] = estado;
414
415 #ifdef DEBUG
416                         switch(dispositivo)
417                         {
418                                 case 1:
419                                         printf("Alarme - Setor 1 (Sensores de Presença) - %d\n",
    estado);
420                                         break;
421
422                                 case 2:
423                                         printf("Alarme - Setor 2 (Sensores de Portas/ Janelas) - %d
    \n", estado);
424                                         break;
425                         }
426                         break;
427 #endif
428         }
429 }
430
431
432 void TrataClienteTCP(int socketCliente, struct sockaddr_in clienteAddr) {
433         int buffer;
434         int tamanhoRecebido;
435         int setor, dispositivo, estado;
436
437         if((tamanhoRecebido = recv(socketCliente, &buffer, sizeof(int), 0)) < 0)
438                 printf("Erro no recv()\n");
439
440
441         setor = buffer/1000;
442         dispositivo = (buffer - (setor*1000))/10;
443         estado = buffer - setor*1000 - dispositivo*10;
444
445         printf("Setor %d, Dispositivo %d, Estado %d\n", setor, dispositivo, estado);
446
447         Log_data(inet_ntoa(clienteAddr.sin_addr), setor, dispositivo, estado);
448         enviaArduino((char)setor, (char)dispositivo, (char)estado, socketCliente, clienteAddr);
449
450 }
451
452 void *server(void *porta)
453 {
454         unsigned short *servidorPorta = (unsigned short *) porta;
455         int servidorSocket;
456         int socketCliente;
457         struct sockaddr_in servidorAddr;
458         struct sockaddr_in clienteAddr;
459         unsigned int clienteLength;
460
461         // Abrir Socket
462         if((servidorSocket = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP)) < 0)
463                 printf("falha no socker do Servidor\n");
464
```

```
465          // Montar a estrutura sockaddr_in
466          memset(&servidorAddr, 0, sizeof(servidorAddr)); // Zerando a estrutura de dados
467          servidorAddr.sin_family = AF_INET;
468          servidorAddr.sin_addr.s_addr = htonl(INADDR_ANY);
469          servidorAddr.sin_port = htons(*servidorPorta);
470
471          // Bind
472          if(bind(servidorSocket, (struct sockaddr *) &servidorAddr, sizeof(servidorAddr)) < 0)
473                  printf("Falha no Bind\n");
474
475          // Listen
476          if(listen(servidorSocket, 10) < 0)
477                  printf("Falha no Listen\n");
478
479          while(1) {
480
481                  printf("Esperando por clientes na porta %d\n", *servidorPorta);
482                  clienteLength = sizeof(clienteAddr);
483                  if((socketCliente = accept(servidorSocket, (struct sockaddr *) &clienteAddr,
     &clienteLength)) < 0)
484                          printf("Falha no Accept\n");
485
486                  printf("Conexão do Cliente %s\n", inet_ntoa(clienteAddr.sin_addr));
487
488                  TrataClienteTCP(socketCliente, clienteAddr);
489                  close(socketCliente);
490
491          }
492          close(servidorSocket);
493  }
494
495  int main(int argc, char *argv[]) {
496
497          pthread_t server_8080;
498          pthread_t server_4000;
499          pthread_t alarm_id;
500
501          Uart_Config(uart0_filestream);
502
503          pthread_create(&server_4000, NULL, &server, &porta_4000);
504          pthread_create(&server_8080, NULL, &server, &porta_8080);
505          pthread_create(&alarm_id, NULL, &Verifica_sensor, NULL);
506
507          pthread_join(server_8080, NULL);
508          pthread_join(server_4000, NULL);
509
510  }
```