

```

1  /* ***** */
2  /* File name:      DroneTimer.cpp */
3  /* File description: Timer interface implementation file */
4  /* Author name:     Giacomo Dollevedo, Gustavo Fernandes */
5  /* Creation date:    18nov2020 */
6  /* Revision date:    18dec2020 */
7  /* ***** */
8
9  #include "DroneTimer.h"
10
11  /*
12  *****
13  */
14  /* Method's name:      initTimer
15  */
16  /* Description:        Initialize hardware timer interruptions
17  */
18  /* Entry parameters:    int frequency -> timer interrupt frequency
19  */
20  /*                    void (*fn)(void) -> function pointer that will be executed
21  */
22  /* Return parameters:    n/a
23  */
24  /*
25  *****
26  */
27  void DroneTimer::initTimer(int frequency, void (*fn)(void))
28  {
29      timerFreq = frequency;
30      /*timerBegin(NúmeroDoTimer, PreScaler, UpCount/DownCount*/
31      /*80 -> clock do timer eh 80 MHz. Entao 80.000.000/80 = 1.000.000s = 1
32      microssegundo*/
33      timer0 = timerBegin(0, 80, true);
34
35      /*timerAttachInterrupt(ObjetoTimer, FuncaoDisparada, TipoDaInterrupcao)*/
36      timerAttachInterrupt(timer0, fn, true);
37      /*timerAlarmWrite(ObjetoTimer, periodoInterrupcao(microssegundos) ,
38      RepetirContagem)*/
39      timerAlarmWrite(timer0, 1000000/frequency, true);
40      Serial.println("Timer Interrupt Attached!");
41  }
42
43  /*
44  *****
45  */
46  /* Method's name:      enableTimer
47  */
48  /* Description:        Enable timer interruptions. */
49  /*
50  */
51  /* Entry parameters:    n/a */
52  /*
53  */

```

```

40  /* Return parameters:      n/a
    */
41  /*
    ****
    */
42  void DroneTimer::enableTimer()
43  {
44      timerAlarmEnable(timer0);
45  }
46
47  /*
    ****
    */
48  /* Method's name:          disableTimer
    */
49  /* Description:            Disable timer interruptions.          */
50  /*
    */
51  /* Entry parameters:      n/a          */
52  /*
    */
53  /* Return parameters:     n/a
    */
54  /*
    ****
    */
55  void DroneTimer::disableTimer()
56  {
57      timerAlarmDisable(timer0);
58  }
59
60
61
62
63  /*NAO USAMOS PRA NADA*/
64  /*
    ****
    */
65  /* Method's name:          setFrequency
    */
66  /* Description:            Set the timer frequency          */
67  /*
    */
68  /* Entry parameters:      int frequency -> frequency to be set    */
69  /*
    */
70  /* Return parameters:     n/a
    */
71  /*
    ****
    */
72  void DroneTimer::setFrequency(int frequency)
73  {
74      timerFreq = frequency;
75  }
76
77  /*NAO USAMOS PRA NADA*/
78  /*
    ****

```

```

*/
79 /* Method's name:          getFrequency
   */
80 /* Description:           Get the timer frequency          */
81 /*
   */
82 /* Entry parameters:      n/a                          */
83 /*
   */
84 /* Return parameters:     int -> Timer frequency
   */
85 /*
   ****
   */
86 int DroneTimer::getFrequency()
87 {
88     return timerFreq;
89 }
90

```