# Código fonte do Relógio Digital

# Funções utilizadas:



# Função Loop()

```
void loop()

{
          ContaTempo();                                  //utiliza o contador interno do arduino
          char key = keypad.getKey();
          if (key != NO_KEY)
{
         if (key == '*') {
         MostrarRelogio = false;
         AcertarHora();
   }
  }
          if (MostrarRelogio) Relogio();                 // mostra as horas ao ligar
}                                                        // fim loop
```

# Função ContaTempo()

```
void ContaTempo()

{                                   // Inicio do Contador do Relógio

        unsigned long tempoAtual = millis();            //Tempo atual em ms
        digitalWrite(ledSegs, ( Seg % 2 == 0) ? 1 : 0);   //pisca leds dos segundos
        if ( tempoAtual - tempoMemo > 1000) {
        tempoMemo = tempoAtual;
        Seg = Seg + 1;                                  //incrementa segundos
 }


 if (Seg > 59)

{

        Seg = 0;
        Min = Min + 1;                                  //incrementa minutos
 }

 if (Min > 59)

{

        Min = 0;
        Hora = Hora + 1;                                //incrementa Horas
 }

 if (Hora > 23)

{

        Hora = 0;
        Min =  0;
        Seg =  0;
 }


}                                   // Fim do Contador do Relógio
```

# Função AcertarHora()

```
void AcertarHora()
{
            int x = 0;
            String pegaHora, pegaMin, novahora;
            digitalWrite(ledConfig, 1);

 do {
          if (novahora.length() > 0) {
          AcertandoHorario(novahora); // vai mostrando os dígitos e traços durante a digitação da nova hora
   } else hora();                                  // mostra  a palavra hora
             char key =  keypad.getKey();
             if ( key != NO_KEY and key != '*' )
   {
               if (key == '#') break;
               novahora = novahora + String(key);
               if (novahora.length() > 3)
    {
                pegaHora = novahora.substring(0, 2);    // pega os 2 dig. da hora.
                Hora = pegaHora.toInt();
                pegaMin  = novahora.substring(2, 4);                  // pega os 2 dígitos dos minutos.
                Min = pegaMin.toInt();
                x = 1;
    }
   }
 } while (x == 0);
            digitalWrite(ledConfig, 0);
            MostrarRelogio = true;
}
```

# Função Relogio()

```
void Relogio() {                                    //Atualiza o Relógio
 if (Hora < 10) {                                   //verifica o 1 digito mais signif. da Hora
   num0();
 }
 else if ((Hora > 9) and (Hora < 20)) {
   num1();
 }
 else if (Hora > 19)num2();

 ApagaTodosDisplays();

 digitalWrite(display1, 0);                         //mostra 1 digito mais signif da Hora
 digitalWrite(display2, 0);
 digitalWrite(display3, 0);
 digitalWrite(display4, 1);
 delay(timev);

 if ((Hora > 9) and (Hora < 20))HoraUnid = Hora - 10;     //verifica o 2 digito menos signif da Hora
 if (Hora >= 20) HoraUnid = Hora - 20;
 if (Hora < 10) HoraUnid = Hora;

 MostraNum(HoraUnid);
 ApagaTodosDisplays();
 digitalWrite(display1, 0);                                 //mostra 2 digito menos signif da Hora
 digitalWrite(display2, 0);
 digitalWrite(display3, 1);
 digitalWrite(display4, 0);

 delay(timev);
```

```
 //Minutos
 if (Min > 9) {                          //verifica 1 digito mais signif dos minutos
   MinDez = int(Min / 10);
 }
 switch (MinDez) {
   case 1: num1(); divisor = 1; break;
   case 2: num2(); divisor = 2; break;
   case 3: num3(); divisor = 3; break;
   case 4: num4(); divisor = 4; break;
   case 5: num5(); divisor = 5; break;
 }

 if (Min < 10)  num0();
 ApagaTodosDisplays();
 digitalWrite(display1, 0);                  //mostra 1 digito mais signif dos minutos
 digitalWrite(display2, 1);
 digitalWrite(display3, 0);
 digitalWrite(display4, 0);
 delay(timev);
 if (Min < 10) MinUnid = Min;                //verifica 2 digito mais signif dos minutos
 if (Min > 9)  MinUnid = Min - (divisor * 10);
 MostraNum(MinUnid);
 ApagaTodosDisplays();                       //mostra 2 digito mais signif dos minutos
 digitalWrite(display1, 1);
 digitalWrite(display2, 0);
 digitalWrite(display3, 0);
 digitalWrite(display4, 0);
 delay(timev);
}
```

# Função MostraNum(int valor)

```
void MostraNum(int valor)  // função para mostrar um digito
{
  switch (valor) {
                case 0: num0(); break;
                case 1: num1(); break;
                case 2: num2(); break;
                case 3: num3(); break;
                case 4: num4(); break;
                case 5: num5(); break;
                case 6: num6(); break;
                case 7: num7(); break;
                case 8: num8(); break;
                case 9: num9(); break;
  }
}
```

# Função ApagaTodosDisplays()

```
void ApagaTodosDisplays()
{
            digitalWrite(display1, 0);
            digitalWrite(display2, 0);
            digitalWrite(display3, 0);
            digitalWrite(display4, 0);
}
```

# Função AcertandoHorario(String valor)

```
void AcertandoHorario(String valor)   // mostra os dígitos sequencialmente quando for digitado a hora
{
        String Dig1, Dig2, Dig3, Dig4;
        Dig1 = valor.substring(0, 1);
        Dig2 = valor.substring(1, 2);
        Dig3 = valor.substring(2, 3);
        Dig4 = valor.substring(3, 4);

 if (valor.length() >= 1) MostraNum(Dig1.toInt()); else traco();

        digitalWrite(display1, 0);
        digitalWrite(display2, 0);
        digitalWrite(display3, 0);
        digitalWrite(display4, 1);
        delay(timev);

 if (valor.length() >= 2) MostraNum(Dig2.toInt()); else traco();

        digitalWrite(display1, 0);
        digitalWrite(display2, 0);
        digitalWrite(display3, 1);
        digitalWrite(display4, 0);
        delay(timev);

 if (valor.length() >= 3)MostraNum(Dig3.toInt()); else traco();

        digitalWrite(display1, 0);
        digitalWrite(display2, 1);
        digitalWrite(display3, 0);
        digitalWrite(display4, 0);
        delay(timev);

 if (valor.length() == 4)MostraNum(Dig4.toInt()); else traco();

        digitalWrite(display1, 1);
        digitalWrite(display2, 0);
        digitalWrite(display3, 0);
        digitalWrite(display4, 0);
        delay(timev);

}
```

# Funções para construir os números

```
void hora() {

 h_dig();

          digitalWrite(display1, 0);
          digitalWrite(display2, 0);
          digitalWrite(display3, 0);
          digitalWrite(display4, 1);
          delay(timev);
 o_dig();

          digitalWrite(display1, 0);
          digitalWrite(display2, 0);
          digitalWrite(display3, 1);
          digitalWrite(display4, 0);
          delay(timev);
 r_dig();

          digitalWrite(display1, 0);
          digitalWrite(display2, 1);
          digitalWrite(display3, 0);
          digitalWrite(display4, 0);
          delay(timev);
 a_dig();

          digitalWrite(display1, 1);
          digitalWrite(display2, 0);
          digitalWrite(display3, 0);
          digitalWrite(display4, 0);
          delay(timev);
}

void h_dig() {

          digitalWrite(segA, 1);
          digitalWrite(segB, 1);
          digitalWrite(segC, 0);
          digitalWrite(segD, 1);
          digitalWrite(segE, 0);
          digitalWrite(segF, 0);
          digitalWrite(segG, 0);
}
```

```
void a_dig() {

        digitalWrite(segA, 0);
        digitalWrite(segB, 0);
        digitalWrite(segC, 0);
        digitalWrite(segD, 1);
        digitalWrite(segE, 0);
        digitalWrite(segF, 0);
        digitalWrite(segG, 0);
}

void r_dig() {

        digitalWrite(segA, 1);
        digitalWrite(segB, 1);
        digitalWrite(segC, 1);
        digitalWrite(segD, 1);
        digitalWrite(segE, 0);
        digitalWrite(segF, 1);
        digitalWrite(segG, 0);
}

void o_dig() {

        digitalWrite(segA, 1);
        digitalWrite(segB, 1);
        digitalWrite(segC, 0);
        digitalWrite(segD, 0);
        digitalWrite(segE, 0);
        digitalWrite(segF, 1);
        digitalWrite(segG, 0);
}

void traco() {

        digitalWrite(segA, HIGH);
        digitalWrite(segB, HIGH);
        digitalWrite(segC, HIGH);
        digitalWrite(segD, HIGH);
        digitalWrite(segE, HIGH);
        digitalWrite(segF, HIGH);
        digitalWrite(segG, LOW);
}

void num1() {

        digitalWrite(segA, HIGH);
        digitalWrite(segB, LOW);
        digitalWrite(segC, LOW);
        digitalWrite(segD, HIGH);
        digitalWrite(segE, HIGH);
        digitalWrite(segF, HIGH);
        digitalWrite(segG, HIGH);
}
```

```
void num2() {

        digitalWrite(segA, 0);
        digitalWrite(segB, 0);
        digitalWrite(segC, 1);
        digitalWrite(segD, 0);
        digitalWrite(segE, 0);
        digitalWrite(segF, 1);
        digitalWrite(segG, 0);
}

void num0() {

        digitalWrite(segA, 0);
        digitalWrite(segB, 0);
        digitalWrite(segC, 0);
        digitalWrite(segD, 0);
        digitalWrite(segE, 0);
        digitalWrite(segF, 0);
        digitalWrite(segG, 1);
}

void num3() {

        digitalWrite(segA, 0);
        digitalWrite(segB, 0);
        digitalWrite(segC, 0);
        digitalWrite(segD, 0);
        digitalWrite(segE, 1);
        digitalWrite(segF, 1);
        digitalWrite(segG, 0);
}

void num4() {

        digitalWrite(segA, 1);
        digitalWrite(segB, 0);
        digitalWrite(segC, 0);
        digitalWrite(segD, 1);
        digitalWrite(segE, 1);
        digitalWrite(segF, 0);
        digitalWrite(segG, 0);
}

void num5() {

        digitalWrite(segA, 0);
        digitalWrite(segB, 1);
        digitalWrite(segC, 0);
        digitalWrite(segD, 0);
        digitalWrite(segE, 1);
        digitalWrite(segF, 0);
        digitalWrite(segG, 0);
}
```

```
void num6() {

        digitalWrite(segA, 0);
        digitalWrite(segB, 1);
        digitalWrite(segC, 0);
        digitalWrite(segD, 0);
        digitalWrite(segE, 0);
        digitalWrite(segF, 0);
        digitalWrite(segG, 0);
}

void num7() {

        digitalWrite(segA, 0);
        digitalWrite(segB, 0);
        digitalWrite(segC, 0);
        digitalWrite(segD, 1);
        digitalWrite(segE, 1);
        digitalWrite(segF, 1);
        digitalWrite(segG, 1);
}

void num8() {

        digitalWrite(segA, 0);
        digitalWrite(segB, 0);
        digitalWrite(segC, 0);
        digitalWrite(segD, 0);
        digitalWrite(segE, 0);
        digitalWrite(segF, 0);
        digitalWrite(segG, 0);
}


void num9() {

        digitalWrite(segA, 0);
        digitalWrite(segB, 0);
        digitalWrite(segC, 0);
        digitalWrite(segD, 0);
        digitalWrite(segE, 1);
        digitalWrite(segF, 0);
        digitalWrite(segG, 0);
}
```