```
5298 //*****************************************************************************
5299 //**  LOOP   *****************************************************************
5300 //***********  LOOP   ********************************************************
5301 //*******************  LOOP   ***************************************************
5302 //*****************************  LOOP   ****************************************
5303 //***********************************  LOOP   ***********************************
5304 //*****************************************************************************
5305 //
5306 //  Name: loop
5307 //
5308 //  Modification date:
5309 //  Changed by:
5310 //
5311 //  Function:
5312 //  - Check result of configuration test
5313 //  - Read all input tags
5314 //  - Start the FSM
5315 //
5316 //*****************************************************************************
5317
5318 void loop()
5319 {
5320
5321   boolean ActionResult;
5322   long positions[1];
5323
5324   //*============================================================================
5325   //*  Standard actions on front of cycle. Don't change.
5326
5327   if ( CConfigurationOK == false )
5328   {
5329     Serial.println("");
5330     Serial.println(F("%FAT-CNFFAILURE-SETUP, Failure, code not executed or stopt."));
5331     delay(2000);
5332     if ( Pin13 == 1 ) {
5333       Pin13 = 0;
5334       digitalWrite (13, Pin13);
5335     }
5336     else {
5337       Pin13 = 1;
5338       digitalWrite (13, Pin13);
5339     }
5340   }
5341   else
5342   { //begin else from the main loop
5343
5344       if (NoInputPins != 0) ReadInputs();
5345       if (NoAnaInPins != 0) ReadAnaIn();
5346       if (Timer(200, TimUS)) if (NoUltrasonic != 0) ReadUltrasonic();
5347       if (Timer(5000, TimTT)) if (NoTemp != 0) ReadTemp();
5348       if (Timer(5000, TimGpsPoll)) Serial2.println("$PUBX,00*33");
5349
5350       //  when a HMI client is connected, it will send a message every 2 seconds, triggering timer
5351       //  TimHMIWtd for 4 seconds again and setting HMIWatchDoog to true. So when there HMIWatchDog is true,
5352       //  check the timer, if it is expired, turn HMIWatchDog to false;
5353       //  In the program you can do what ever you want with the boolean HMIWatchDog. Up to you.
```

```
5354        if (HMIWatchDog == true)
5355        {
5356            if (Timer(10000, TimHMIWtd)) HMIWatchDog = false;
5357            if (Pin13 == 0)
5358            {
5359                if (Timer(260, TimBlinkWtdOn)) digitalWrite(13, 1);
5360                if (Timer(280, TimBlinkWtdOff))
5361                {
5362                    digitalWrite(13, 0);
5363                    CancelTimer(TimBlinkWtdOn);
5364                }
5365            }
5366        };
5367
5368
5369        // when flashing 500mS (output 13), the configuration is OK
5370        if (Timer(Blinking, TimBlinking))
5371        {
5372            if (Pin13 == 1)
5373            {
5374                Pin13 = 0;
5375                digitalWrite(13, Pin13);
5376            }
5377            else
5378            {
5379                Pin13 = 1;
5380                digitalWrite(13, Pin13);
5381            }
5382        };
5383
5384        if (Timer(60000, TimISR))
5385        {
5386            //To prevent ISR_count to change intermediate bij the ISR function, interrups will be blocked to make a copy of ISR_Count
5387            noInterrupts();
5388            ISR_Count_Copy = ISR_Count;
5389            //... and enabled again
5390            interrupts();
5391            //Serial.println ( "ISR called for " + String (ISR_Count_Copy) + " times");
5392        }
5393
5394        if (DelayGMI)
5395        {
5396            // If DelayHMI is true, the HMI messages are turned off.
5397            // Wait for a certain time and turn the messages on again.
5398            // This is done to allow HMI to process the GMI messages.
5399            if (Timer(1500, TimDelayGMI))
5400            {
5401                UseHMISerial = true;
5402                DelayGMI = false;
5403            }
5404        }
5405
5406        MachineState = CurrentState();
5407        do
5408        {
5409
```

```
5410
5411          //*  End of Standard actions on front of cycle
5412          //*=============================================================================================
5413
5414
5415
5416          //*=============================================================================================
5417          //*  Application code
5418          //*
5419          //*  Name: UserFSM
5420          //*  Version:
5421          //*  Date:
5422          //*  Author:
5423          //*
5424          //*  Short description:
5425          //*
5426          //*
5427          //*=============================================================================================
5428          //*=============================================================================================
5429          //*============ START USER APPLICATION ======= START USER APPLICATION ====================
5430          //*=============================================================================================
5431          //*=============================================================================================
5432          //
5433          // State machine
5434          //
5435          // UserFSM contains the FSM software.
5436          // Each phase exists of 2 parts, e.g. the phase actions and transition conditions.
5437          // In the phase actions, all activated tags are mentioned. Deactivation is not needed, deactivation is
5438          // done at the end of the main loop. In the conditions section, all conditions for the transitions to another
5439          // phase are mentioned. When all condition are true, the new phase can be set. It is possible to programm
5440          // more then 1 transition. In such a case it is necassery to keep the priority in mind.
5441          //
5442
5443
5444          UserFSM();
5445
5446
5447          //*=============================================================================================
5448          //*=============================================================================================
5449          //*============  END USER APPLICATION ========= END USER APPLICATION ====================
5450          //*=============================================================================================
5451          //*=============================================================================================
5452
5453          //*  Standard actions on the end of cycle. Don't change.
5454
5455        MachineState = CurrentState();
5456      } while ( MachineState != 0 );
5457
5458      ComToRunState();
5459
5460
5461      if ( CycleCalc == true ) CalcCycleCount();
5462
5463      // Outputs mentioned in the phases are set. Digitals not activated are reset, unchanged analogs and servos keep there value!
5464      SetOutputs();
5465      SetAnaOutputs();
```

```
5466      SetServos();
5467      //SetMarkerText();
5468
5469      // Is there inputdata on the serial monitor available?
5470      if (Serial.available())
5471        {
5472          delay(10);
5473          SerMonitorCommand();
5474        }
5475
5476      // Serial2 used for GPS....(?)
5477      if (Serial2.available()) HandlingGPSSerial();
5478
5479      // Is there inputdata available on the 2e serial line (Supervisor channel)?
5480      if ((UseHMISerial) || (DelayGMI))
5481        {
5482          if (Serial1.available()) HandlingHMISerial();
5483          HMIFromQueue();
5484        }
5485
5486    } //end else main loop after okee configuration
5487  }
5488
5489
5490
```