Application state diagram

Datasheet Application

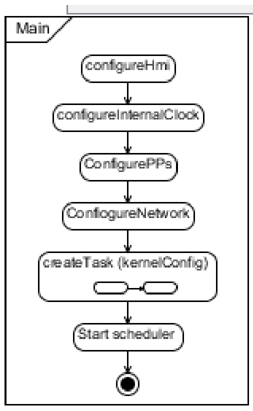
This document will show how the program works.

Baptiste MORAND

6/28/2016

Contents

I. Main Function 2
  A. Description 2
  B. Function 2
II. Task create 3
  A. Description 3
  B. Function 3
III. TimeProtocolTask 3
  A. Task 3
    1. Description 3
    2. Function 3
  B. Sender 4
    1. Description 4
    2. Function 4
  C. Receiver 5
    1. Description 5
    2. Function 5
  D. Correction 6
    1. Description 6
    2. Function 6
  E. Correct Offset 7
    1. Description 7
    2. Function 7
IV. HMI Task 7
  A. Description 8
  B. Function 8
V. Interrupt 8
  A. PPS 8
    1. Description 8
  B. Internal Timer 9
    1. Description 9



I. Main Function

`int main (void) (main.cpp)`

A. Description

Call at the beginning of the application, configure peripherals create the task and run the scheduler

B. Function

Hmi : void configureHMI(void) (hmi.h)

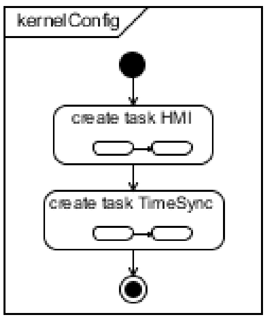
InternalClock : void configureInternalClock(void)(internalClock.h)

Pps : void configurationPPS(void) (ppsGPS.h)

Network : void configurationNetwork(void) (network.h)

Create Task : void kernelConfig(void) (utask.h)

Start Scheduler : void vTaskStartScheduler(void) (task.h)



II. Task create

void kernelConfig(void) (utask.h)

A. Description

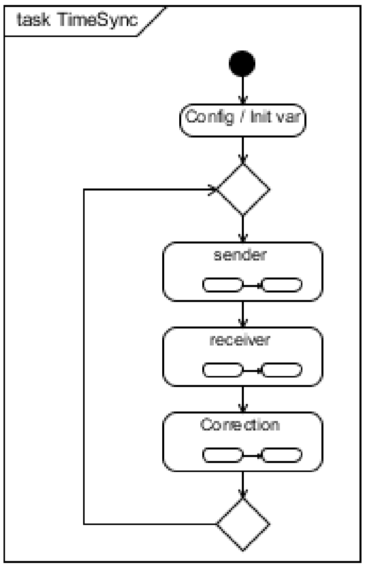
Create the task, the task will be manage by the scheduler :

- Task HMI is priority 1 (low)
- Task TimeProtocol is priority 3 (high)

B. Function

Hmi task : void HMITask(void) (hmi.h)

Time Protocol task : void timeProtocolTask(void) (timeProtocol.h)



III. TimeProtocolTask

A. Task

1. Description

This task will be running in a loop, and consist in 3 function to sending receiving and make the correction.

2. Function

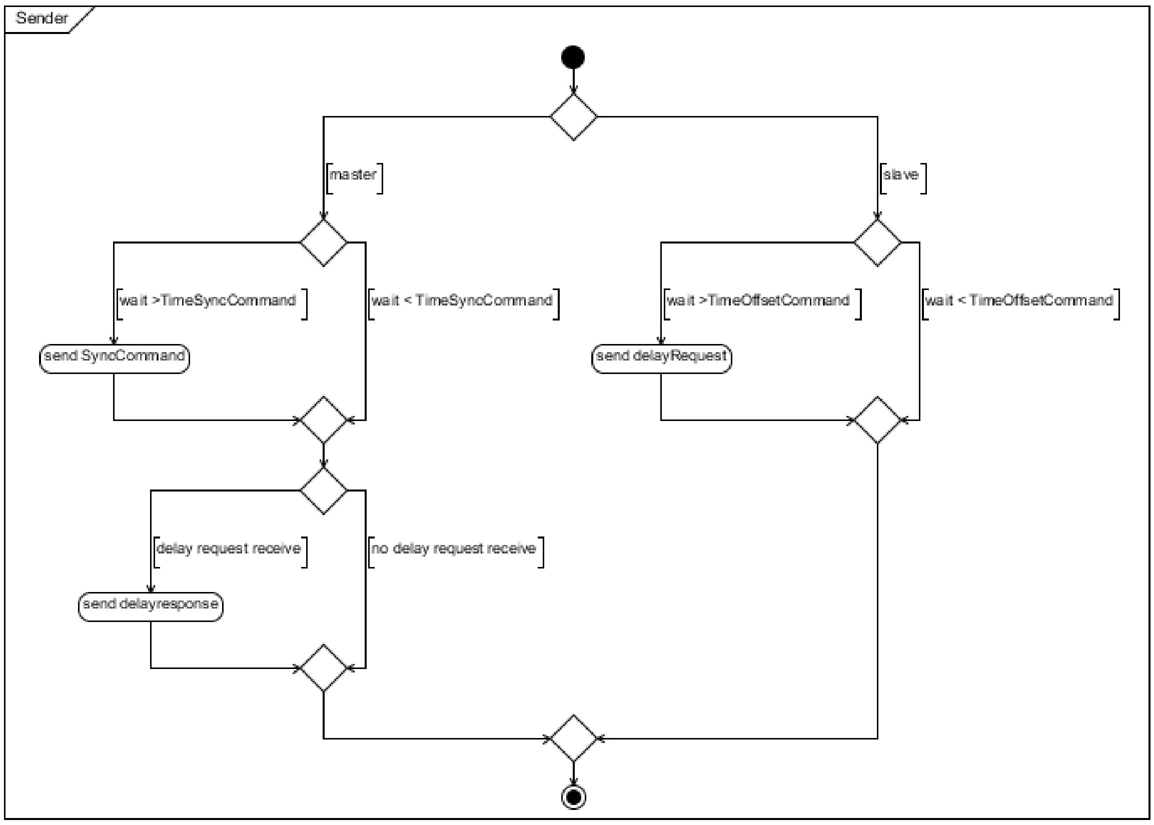
Function receiving : void receiver(void) (timeProtocol.h)

Function sending : void sender(void) (timeProtocol.h)

Function correction: void correction(void) (timeProtocol.h)

B. Sender

void sender(void) (timeProtocol.h)



1. Description

In function of the node it will send the correct command, all command are trigger by a time or an event.

Trigger event :

- **Sync Request :** time define in conf_time_protocol.h manage by the time of the RTOS
- **DelayRequest :** time define in conf_time_protocol.h manage by the time of the RTOS
- **Delay Response :** event declare if timeRequest is receive

2. Function

Send Sync command : void sync(void) (timeProtocol.h)

Send Delay Request : void delayRequest(void) (timeProtocol.h)

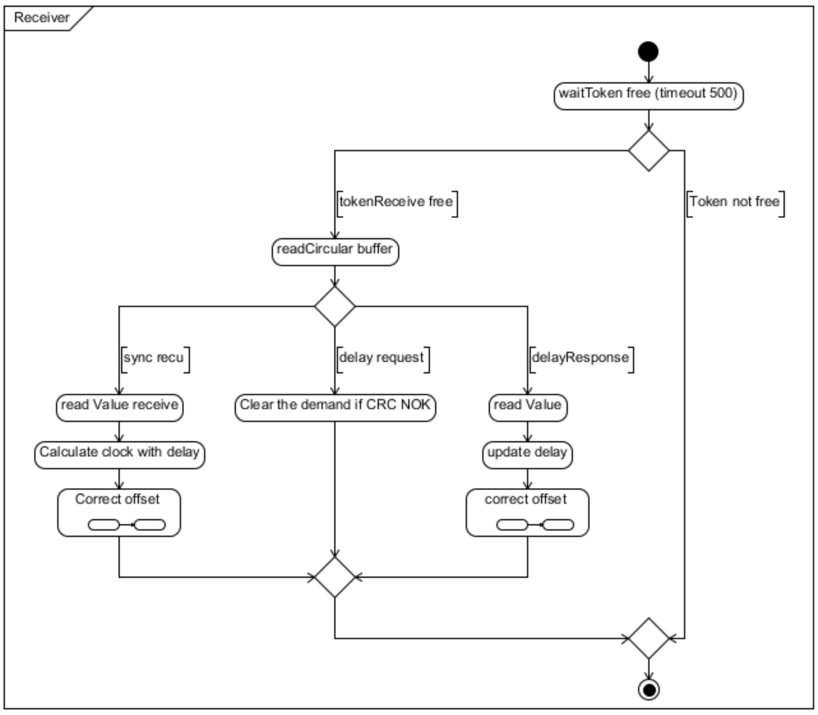
Send Delay Response : void delayResponse(uint8_t id) (timeProtocol.h) with id the id of the slave

Time sync : #define TIMESYNC (conf_timeProtocol.h)

Time Delay Request : #define TIMEDELAYREQUEST (conf_timeProtocol.h)

C. Receiver

void receiver(void) (timeProtocol.h)



1. Description

The token will be free in an interrupt to synchronise the task with the reception (http://www.freertos.org/binary-semaphore.gif).

After he will parse the command and decide what he have to do.

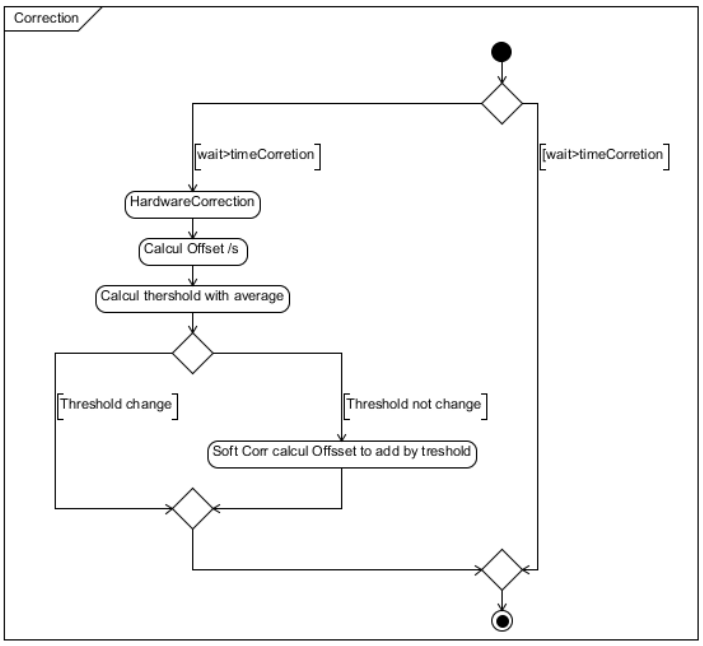
2. Function

All the code are in void receiver(void) (timeProtocol.h)

Function correctOffset : `void updateClock(void)` (timeProtocol.h) the current offset is a global value. `(timeprot.offset)`

D. Correction

void correction(void) (timeProtocol.h)



1. Description

Correction is trigger by a time or an event.

The programme have 2 things to correct :

- Hardware Correction : Hardware correction will change the threshold see part internalClock.
- Software Correction will correct the clock every n threshold .

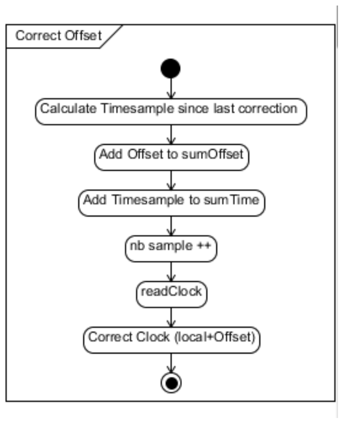
If Hardware Correction is made you have to disable software correction up to the next correction.

2. Function

Time Correction : `#define TIME_CORRECTION` (conf_timeProtocol.h)

E. Correct Offset

```
void updateClock(void)
```


Correct Offset
Calculate Timesample since last correction
Add Offset to sumOffset
Add Timesample to sum Time
nb sample ++
readClock
Correct Clock (local+Offset)

1. Description

For the correction he will save the offset and the timeSample. And after he will correct his clock.

2. Function

Read clock : `uint32_t readClock(Clock* timeClock) (internalClock.h)`

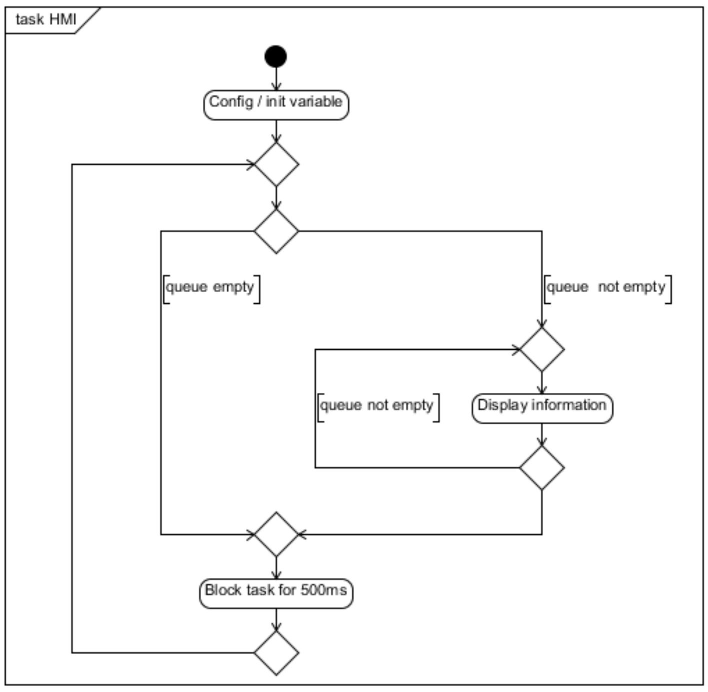
`Sum Offset` `: sumOffset (timeProtocol.h)`

`Sum Time` `: timeProt.correction.sumTime(timeProtocol.h)`

`Nb sample` `: timeProt.correction.nbCorrection (timeProtocol.h)`

IV. HMI Task

void HMITask(void) (hmi.h)



A. Description

The HMI send to a UART communication the data in the queue.

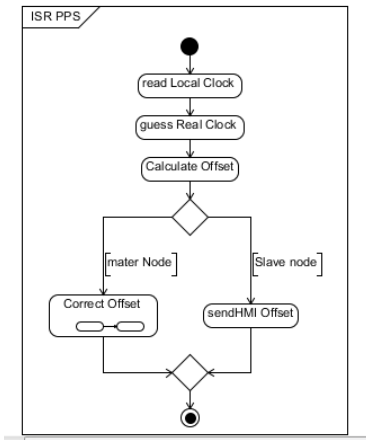
B. Function

HMI queue : `xQueueHandle uartQueue (hmi.h)`

V. Interrupt

A. PPS

```
void ppsISR(void)
```

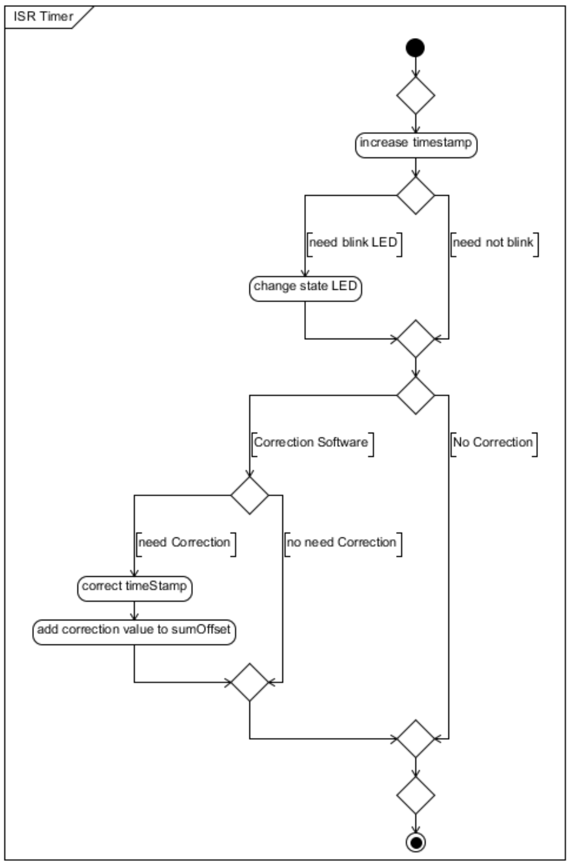


1. Description

PPS will calculate the offset and print it for a Slave node or correct it if it is a master node.

B. Internal Timer

```
void isrInternalClok(void )
```



1. Description

This interrupt will manage the led, and the timestamp. If there is a Software correction he will add the correction to the timestamp.