#include "Thread.h"

Thread::Thread(void (\*callback)(void), unsigned long \_interval){

enabled = true;

onRun(callback);

\_cached\_next\_run = 0;

last\_run = millis();

ThreadID = (int)this;

#ifdef USE\_THREAD\_NAMES

ThreadName = "Thread ";

ThreadName = ThreadName + ThreadID;

#endif

setInterval(\_interval);

};

void Thread::runned(unsigned long time){

// Saves last\_run

last\_run = time;

// Cache next run

\_cached\_next\_run = last\_run + interval;

}

void Thread::setInterval(unsigned long \_interval){

// Save interval

interval = \_interval;

// Cache the next run based on the last\_run

\_cached\_next\_run = last\_run + interval;

}

bool Thread::shouldRun(unsigned long time){

// If the "sign" bit is set the signed difference would be negative

bool time\_remaining = (time - \_cached\_next\_run) & 0x80000000;

// Exceeded the time limit, AND is enabled? Then should run...

return !time\_remaining && enabled;

}

void Thread::onRun(void (\*callback)(void)){

\_onRun = callback;

}

void Thread::run(){

if(\_onRun != NULL)

\_onRun();

// Update last\_run and \_cached\_next\_run

runned();

}