Система за отчитане на производителността на труда в реално време, базирана на контролер ESP32

С. Йорданов, Г. Михалев

System for Automatic Reporting a Labour Productivity in Real Time Based on ESP32 Controller

S. Yordanov, G. Mihalev

Key Words: *Internet of Things (IoT); Wi-Fi module ESP32; Arduino IDE development environment; LPWAN technologies; electronic accounting technologies.*

Abstract. *In this paper, an ESP32 application in implementing systems for automated reporting of the utilization rate of production capacities in real time is considered. A technical solution of remote control and monitoring of the production process in a textile company, through a dedicated module for sending the time spent on a remote server via Wi-Fi, Lan or Bluetooth is proposed. The obtained information is used to control the production process in real time and determination of the remuneration of workers.*

Увод

В съвременните условия при растяща несигурност и висока степен на риск нараства необходимостта от вземане на мотивирани финансово-икономически решения от страна на собствениците и мениджърите на фирмата. Ето защо ефективното управление на ресурсите е особено актуално в настоящия момент. Повишаването на производителността на труда е основен фактор за подобряване на икономическите показатели, финансовото състояние и ефективността на производствено-стопанската дейност на фирмата. От друга страна, производителността на труда е съставен икономически показател, който съпоставя постигнатия резултат (новосъздаден продукт) с вложения трудов фактор при осъществяването на определена икономическа активност на определена икономическа територия за определен период [1]. Един от компонентите му е оптималното използване на машините и съоръженията в процеса на производство. Използването на производствената мощност, както и използването на дълготрайните материални активи се характеризират основно с три показателя [2]:

- **Коефициент за екстензивно използване или използване на производствената мощност по време.** Той представлява отношението на фактическия фонд отработено време към ефективния фонд работно време.
- **Коефициент за интензивно използване на производствената мощност.** Той характеризира използването на производствената мощност в единица време и се определя като отношение на фактически изпълнената работа за единица отработено време към проектната техническа норма.
- **Показатели за интегрално използване.** Характеризират най-пълно използването на производствените мощности. Определят се като произведение на коефициента за екстензивно използване и коефициента на интензивно използване.

Особен интерес за мениджърите на предприятията представлява коефициентът за интензивно използване на производствените мощности. Той носи информация както за реалното натоварване на производствените ресурси, така и за състоянието на технологичното оборудване, трудовата дисциплина, материално-техническата снабденост на предприятието и не на последно място – квалификацията на обслужващия персонал. Целта на настоящата разработка е проектирането и изграждането на интегрирана апаратно-програмна система, предназначена за отчитане на натоварването на производствените мощности в реално време. Разработката е следствие на техническо задание, възложено от текстилна фирма.

Постановка на задачата

Системата за мониторинг е разработката на базата на техническо задание със следните основни пункта:

- Разработената на автономна интегрирана система, отчитаща работното състояние на текстилни машини.
- Информация за състоянието на машините се получава от сензори, отчитащи импулси, или от on/off сензори.
- Данните за отработеното време да се съхраняват на отдалечен сървър.
- Връзката на контролерите с базата от данни да бъде през Ethernet или WiFi с възможност за промяна на настройките за връзка с мрежата.
- Да има възможност за разширение чрез добавяне на допълнителни контролери и машини към все-

ки контролер (до седем машини на контролер) и допълнителни.

- При отпадане на захранването или връзката със сървъра данните от измерванията да се съхраняват и изпратят до сървъра в подходящ момент.
- Наличие на специализиран софтуер за обработка и визуализация на получените входни данни от всяка текстилна машина, изискванията към които са:

– Визуализация за коефициент на използване на машината (КИМ) и отработеното време за дефиниран период (работна смяна). Периодът на работните смени е дефиниран предварително от администратора на системата. Вариантите са два – сменен режим (три смени в денонощие) и редовна смяна. Да може да се избира типът работа (сменен режим или редовна смяна) за всяка машина поотделно.

– Запазване на статистика и възможност за търсене и визуализация за КИМ-а на всяка машина чрез задаване на времеви период.

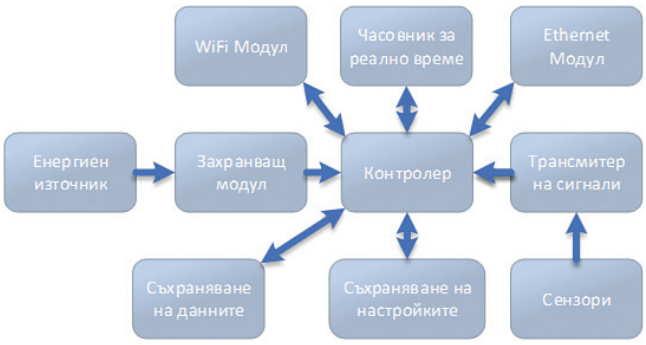
– Възможност за автоматично изпращане на e-mail, съдържащ дневна, седмична или месечна справка за КИМ.

– Възможност за автоматично изпращане на e-mail при загуба и възстановяване на комуникацията с контролерите.

– Възможност за добавяне на допълнителни контролери към системата и други.

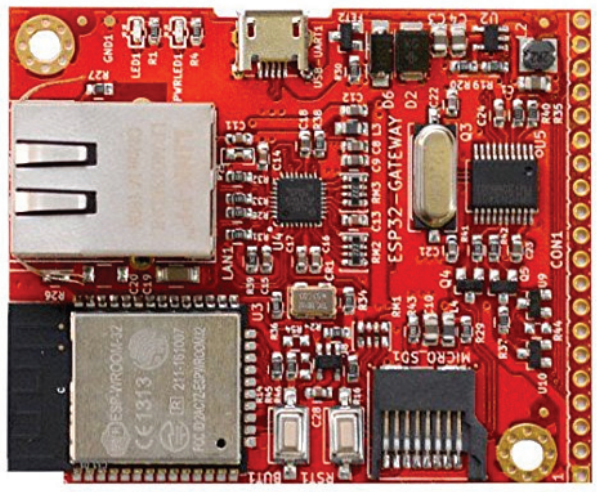
Разработване на системна архитектура, подходяща за мрежово свързани смарт устройства

Анализът на заданието показва, че за реализирането му е необходимо използването на множество свързани в мрежа смарт устройства притежаващи следните компоненти: Ethernet или WiFi модул, часовник за реално време (RTC), място за съхраняване на данните, например SD карта, място за съхраняване на настройките, галванично разделени минимум пет цифрови входа. Един обобщен блоков модел на смарт устройството е показан на *фиг. 1*.



Фиг. 1. Обобщен модел на мрежово свързано смарт устройство

Заданието може да бъде реализирано по два начина: реализация, базирана на PLC, и такава, базирана на микропроцесорна система. От икономическа гледна точка реализацията на базата на микропроцесорна система е многократно по-изгодно. На пазара съществуват множество микропроцесорни системи и едноплаткови компютри като Raspberry Pi, Orange Pi, Rock64, Asus Tinker Board, Pine A64-LTS`, Single Board Computer S5P4418, NanoPC-T3 Octa-core Cortex A53 SBC Sells, Sparky SBC, STM32F407IGT6 STM32 Cortex-M4 Development Core Board , ESP-WROVER-KIT V3 Getting Started Guide, WiDo – An Arduino Compatible IoT Board и други [8,9,10,11,12,13,14,15], на които може да се стъпи при реализацията на системата.



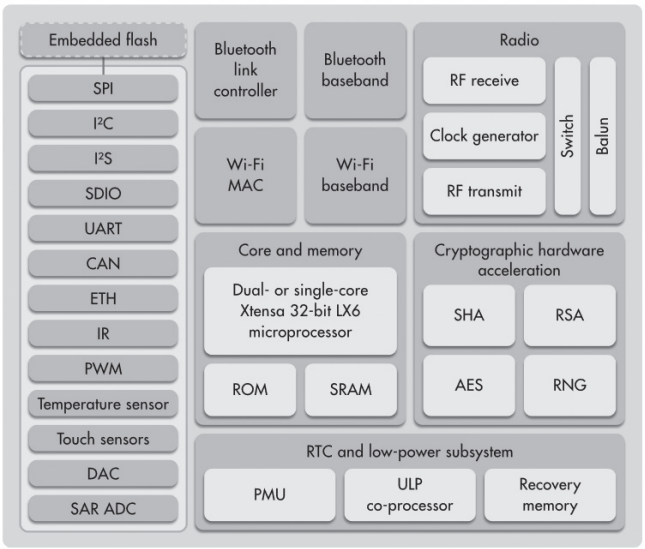
Фиг. 2 . Развойна система Olimex ESP32-GATEWAY

При направения инженерно-икономически анализ основан на критериите: функционални възможности на контролера; наличие на Ethernet и WiFi интерфейс; надеждност; възможност за съхраняване за продължителен период на измерените данни; нужда от минимална доработка на хардуера; цена и други, беше избрана развойната система ESP32-GATEWAY development board with WiFi BLE Ethernet, micro SD card UEXT and GPIO [17], производство на фирмата OLIMEX Ltd. Пловдив (*фиг. 2*).

Фиг. 3. Wi-Fi модул ESP-WROOM-32

Развойната система е базирана на микроконтролера ESP-WROOM-32 (фиг. 3). ESP32 е серия от нискоенергийни микроконтролери с ниска цена. Той представлява еднокристална система с интегрирани Wi-Fi и Bluetooth контролери и вградена антена в него [16]. Освен процесора развойната система включва Ethernet интерфейс, Micro USB конектор и Micro SD карта. На куплунг са изведени 20 GPIO пина на контролера. ESP-WROOM-32 е мощен, универсален WiFi-bt-ble MCU модул, който е насочен към широк спектър от приложения – от нискочестотни сензорни мрежи до най-взискателните задачи, например кодиране на глас, поточно предаване на музика и MP3 декодиране и други. Също така намира приложение при експериментиране и изграждане на Internet of Things (IoT) [3], проекти и прототипи. В серията ESP32, чийто представител е ESP-WROOM-32, е вграден двуядрен 32-битов микропроцесор Xtensa LX6 160-240MHz с 600 DMIPS; 448 KBytes ROM, 520 KBytes вградена SRAM, RTC таймер с 16 KB SRAM памет,

достъпна от процесора; 4 MB Flash памет. ESP-WROOM-32 има 38 пина, от които 34 са програмируеми GPIO порта на 3.3 V. Контролерът поддържа 32 вектора на прекъсване от около 70 източника на прекъсване. В чипа е интегриран WiFi 802.11n 2.4 GHz предавател с максимална скорост от 150 Mbps при изходна мощност на антената от 22 dBm, поддържащ следните стандарти (FCC/CE/IC/TELEC/KCC/SRRC/NCC) и режими (Station/SoftAP/SoftAP+Station/ P2P). Вграден двурежимен Bluetooth v4.2 (EDR и BLE). Връзката с контролера може да се осъществи през интерфейсите SPI, I2C,I2S, UART, IR, CAN 2.0 и други. Вътрешната архитектура на контролера е показана на *фиг. 4*.

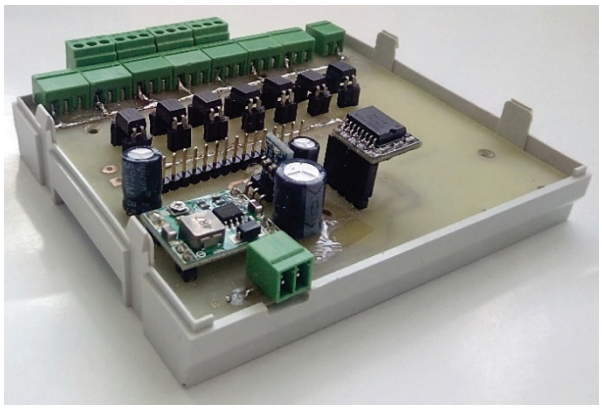


Фиг. 4. Функционална блок диаграма на ESP-WROOM-32

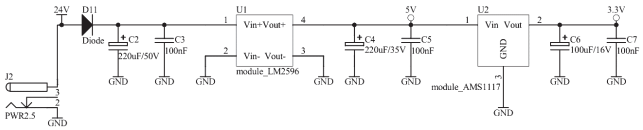
Ядрата на процесора могат да се управляват поотделно, а тактовата им честота може да се регулира в диапазона от 80 MHz до 240 MHz. Потребителят може да изключи процесора и да използва копроцесор с ниска консумация за постоянно наблюдение на периферията за промяна или наличие на прагова стойност. Консумираният ток от чипа е по-малък от 5 μA, което го прави подходящ за вграждане в устройства, захранвани от батерии. В софтуерно отношение платформата е силно развита и има доста голям набор от езици за програмиране като C/C++, Arduino C, Lua, MicroPython и Espruino. Освен основният системен софтуер на Espresiff, който е AT базиран (сериен с AT команди), са налични други методи за програмиране. На ниско ниво SDK на Espressif – в два варианта Non OS и Free RTOS. За запознатите с Arduino – има разработена поддръжка в Arduino IDE за директно програмиране на ESP32, която е използвана при програмирането на тази разработка.

Схемно решение

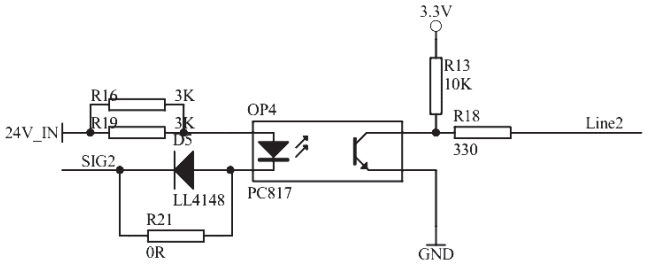
В развойна система Olimex ESP32-GATEWAY притежава пет от елементите, включени в обобщения модел на мрежово свързаното смарт устройство. За реализиране на пълната функционалност на устройството е разработена допълнителна разширителна интерфейсна платка (*фиг. 5*). Схемите на свързване на интерфейсна платка са показани на *фиг. 6*.



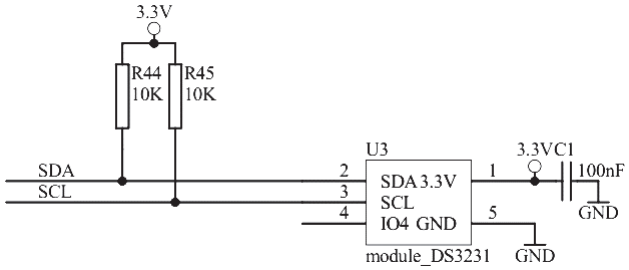
Фиг. 5. Разширителна интерфейсна платка



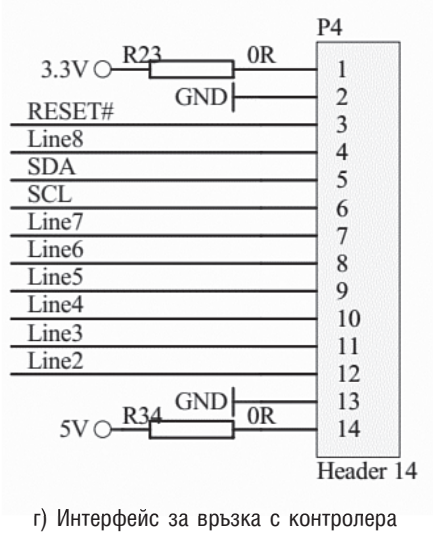
а) Захранващ модул



б) Модул за галванично разделяне на сензорите



в) Свързване на часовник за реално време



г) Интерфейс за връзка с контролера

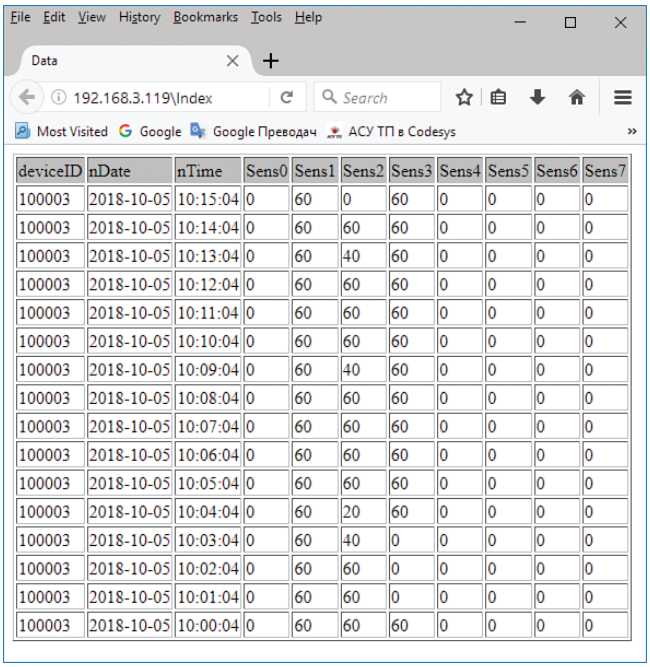
Фиг. 6. Принципна схема на разширителната платка

При реализирането на разширителната платка са използвани модулите: DS3231(часовник за реално време); понижаващ DC-DC регулируем конвертор на напрежение с входно напрежение 4.5 V ~ 28 V и изходно напрежение 0.8 ~ 20 V LM2596; понижаващ DC-DC конвертор на напрежение 5 V-3.3 V AMS1117. Галваничното разделяне на сензорите е реализирано с оптрони PC817.

Функциониране на системата

Разработената система е изградена на три нива: **ниво контролери, сървърно ниво и конфигуриращ софтуер**.

Първото ниво представлява мрежа от смарт контролери, събиращи информация за отработеното време на машините. На всяка минута контролерите изпращат информация за отработеното време от машините към базата с данни, разположена на отдалечен сървър. Връзката със сървъра се осъществява по Ethernet и WiFi мрежа, като е възможна едновременната работа на двете мрежи. Ако са включени и двете мрежи, данните до базата данни се предават по по-сигурната жична линия Ethernet. В тази ситуация WiFi мрежата се използва за параметриране на контролера и за следене на състоянието на процесите, свързани към контролера чрез локална WEB страница (*фиг. 7*). В тази страница се визуализира информацията за работата на машините за последните 30 минути.



deviceID	nDate	nTime	Sens0	Sens1	Sens2	Sens3	Sens4	Sens5	Sens6	Sens7
100003	2018-10-05	10:15:04	0	60	0	60	0	0	0	0
100003	2018-10-05	10:14:04	0	60	60	60	0	0	0	0
100003	2018-10-05	10:13:04	0	60	40	60	0	0	0	0
100003	2018-10-05	10:12:04	0	60	60	60	0	0	0	0
100003	2018-10-05	10:11:04	0	60	60	60	0	0	0	0
100003	2018-10-05	10:10:04	0	60	60	60	0	0	0	0
100003	2018-10-05	10:09:04	0	60	40	60	0	0	0	0
100003	2018-10-05	10:08:04	0	60	60	60	0	0	0	0
100003	2018-10-05	10:07:04	0	60	60	60	0	0	0	0
100003	2018-10-05	10:06:04	0	60	60	60	0	0	0	0
100003	2018-10-05	10:05:04	0	60	60	60	0	0	0	0
100003	2018-10-05	10:04:04	0	60	20	60	0	0	0	0
100003	2018-10-05	10:03:04	0	60	40	0	0	0	0	0
100003	2018-10-05	10:02:04	0	60	60	0	0	0	0	0
100003	2018-10-05	10:01:04	0	60	60	0	0	0	0	0
100003	2018-10-05	10:00:04	0	60	60	60	0	0	0	0

Фиг. 7. Локално Web приложение

Времето за визуализация на данните е съобразено с буфера на цикличната опашка, отговаряща за изпращане на данните към сървъра. Параметрирането на контролерите се извършва през UART или UDP посредством командни низове от вида !WIP:192.168.1.152. На всяка минута данните от измерванията се съхраняват във файлове, разположени на micro MMC карта памет, и се изпращат към базата с данни на сървъра. Данните от файлове, разположени на SD картата, се достъпват автоматично от SQL сървъра посредством съхранена процедура или постъпили запитвания по UDP, или UART, или през FTP сървър, изграден на контролера.

Софтуерно осигуряване на контролера

За програмирането на Wi-Fi модула във основа на чипа ESP32 е използвана средата за разработване Arduino IDE [4] и тулчейна Arduino ESP master, който може да бъде свален от сайта https://github.com/espressif/arduino-esp32. Ардуино тулчейнът е изграден върху Espressif IoT Development Framework. Фреймуъркът е набор от библиотеки с отворен код и инструменти, предназначени да улеснят внедряването на приложения, работещи под FreeRTOS, работеща върху ESP32 платформа. Този подход позволява да се пишат програми (sketch), като се използват готови функции и библиотеки за Arduino, които се стартират директно на ESP32 без наличие на външен Arduino контролер. В пакета са включени библиотеки, позволяващи чрез Wi-Fi интерфейс с помощта на протоколите IP, TCP, UDP да се обменят данни с WEB, SSDP, mDNS и DNS сървъри. Използва се flash памет за създаване на файлова система. Поддържа SD карти, сервозадвижвания, управление на периферни устройства по шините SPI и I2C. За модулите ESP8266 и ESP32 има много развит свободен софтуер; например за управление на устройства от смартфони, базирани на IoS и Android, може да се използва облачната услуга Blynk [5].

Разработеният софтуер за управление на контролера работи в следната последователност:

1. Инициализират се параметрите на контролера, като данните за тях се прочитат от EEPROM. При инициализацията се използва специално създадена за целта библиотека от 22 функции. Една от функциите, реализираща четене на байт, е със следния код:

```
uint8_t EEPROM_ReadByte(uint16_t nAddr) {
   uint8_t val;
   EEPROM.begin(EEPROM_SIZE);
   val = EEPROM.read(nAddr);
   EEPROM.end();
   return val;
}
```

2. Инициализират се таймерите и флаговете, свързани с тях. За инициализиране на таймера в ESP32 се изпълняват няколко действия в определена последователност, обединени във функцията `initTimers()`.

```
void initTimers() {
 timerSemaphore=xSemaphoreCreateBinary();
 timer = timerBegin(0, 80, true);
 timerAttachInterrupt(timer, &onTimer, true);
 timerAlarmWrite(timer, 1000, true);
 timerAlarmEnable(timer);
 runWatchdog(); // Run Watchdog Timer
}
```

3. Инициализират се входно изходните портове.

Процесът на инициализация се състои в задаване ре-

жима на пина (вход или изход). Включване на вътрешния pullup резистор и обвързване на пина с функция за обработка на прекъсване. Тази функция се активира по падащ фронт на сигнала, подаван към пина.

```
pinMode(pin, INPUT_PULLUP);
attachInterrupt(digitalPinToInterrupt(pin),
                interruptFunction, FALLING);
```

4. Инициализация на серийния порт. Стандартна процедура, наследена от Arduino.

5. Инициализация на часовника за реално време (RTC). За улеснение на работата с RTC е разработена библиотека от функции, една от които е предназначена за сверяване на часовника от интернет adjustLocalTime(). Тази функция има следния синтаксис:

```
bool adjustLocalTime(){
   struct tm timeinfo;
   setenv("TZ", "UTC+2", 2);
   tzset();
   configTime (2*3600,2*3600,"pool.ntp.org");
 if(!getLocalTime(&timeinfo,2000)){
    return false;}
   return true;
}
```

6. Инициализация на LAN контролера и WiFi.

Разработеният софтуер позволява едновременно функциониране на двете мрежи. За целта на двата контролера се задават различни IP адреси. Назначават се UDP портове и функция, обработваща събитията относно комуникацията. WiFi мрежата се настройва да работи със статичен IP. С това се цели повишаване на сигурността и стабилността на работата.

Инициализацията на WiFi има приблизително следния код:

```
WiFi.enableSTA(true);
if(!WiFi.config(localIP,gateway,subnet,
    primDNS, secDNS)) { return false; }
WiFi.mode(WIFI_STA);
WiFi.onEvent( WiFiEvent );
WiFi.begin(ssid, password);
while((WiFi.status()!=WL_CONNECTED)&&count>0))
{ debugPrint("."); count--; delay(100); }
if( WiFi.status() == WL_CONNECTED ) {
    initUDP( udpWiFiPort );
    WiFi.setAutoReconnect( true );
    return true;
}
```

Контролерите могат да бъдат параметрирани през UART и UDP. За да се определи по коя мрежа е дошла заявката по UDP, на всяка линия се назначава отделен порт.

7. Инициализира се модулът за работа със SD картата. Системата използва micro MMC карта памет с размер от 8GB. В тази карта се съхраняват данните за състоянието на седем машини по минути. Обемът на картата позволява съхраняването на тренда за период от 40 години. Форматът на картата е FAT32. За управлението на данните е разработена библиотека от функции за работа с директории и файлове. Данните за работата на машините се съхраняват в бинарни файлове с пряк достъп за всеки ден. Един запис от файла носи информация за отработеното време за всеки канал в секунди, абсолютното време за отчитане и информация дали данните са прехвърлени в SQL сървъра. Данните от тези файлове се достъпват автоматично от SQL сървъра посредством съхранена процедура или постъпили запитвания по UDP или UART. За улесняване на работата с данните от SD картата на контролера е стартиран FTP сървър, позволяващ изтегляне на файловете. Четенето на един запис от файла се извършва от функцията readRecordFromFile, която има следния синтаксис:

```
int8_t readRecordFromFile(
    TRecord *rec,int offs ){
 if(!workFS.available())
    return SD_FileLessData;
 if( offset >= 0 ) {
    workFS.seek(offs*sizeof(TRecord),SeekSet);}
 workFS.read((byte *)rec,sizeof(TRecord));
 return SD_OK;
}
```

8. Инициализиране и свързване с SQL сървъра. Както беше казано по-горе, данните от измерванията се изпращат на всяка минута към база с данни. Базата с данни е инсталирана на виртуална машина на реален сървър. Използва се MySQL 5.3 за реализация. Добавянето на запис, иницииран от контролерите в базата, задейства тригери и съхранени процедури, които оценяват данните и попълват автоматично таблици с информация за работата по смени на отделните машини. При отсъствие на връзка с даден контролер за период, по-голям от 5 минути, се генерира заявка и се изпраща e-mail до определена група лица, описани в таблица users. Освен това базата данни в определен период от време генерира дневен, седмичен и месечен отчет за работата на машините. Месечният отчет се генерира веднъж на първо число, а седмичният – всеки понеделник. Данните, които се изпращат от контролера към сървъра, се записват в циклична опашка с размер от 30 записа. За самото изпращане на данни се грижи нишка, работеща на нулево ядро на процесора. Така се дава възможност на процесора да следи и обработва информацията за състоянието на сензорите, без да се бави със задачата за осигуряване на комуникацията. Процесорът ESP32 е достатъчно мощен и позволява едновременна работа на няколко задачи. Така например в реализираната система се използват няколко нишки: за обработване на прекъсванията от сензорите; за изпращане на данни към сървъра и други. Пример за създаването на нишка, работеща на ядро 0, е показан в следващия код:

```
void createQueueTasck() {
 int taskCore = 0;
 xHandleAddSQL = NULL;
 xTaskCreatePinnedToCore(
   AddRecordToMySQL, "AddRecordToMySQL",
   6000,NULL,8,&xHandleAddSQL,taskCore);
}
```

Второто ниво (Система за управление на базата с данни) на системата е разположено на отдалечен сървър.

На него е инсталирана релационна база от данни, в която се съхраняват данните от измерванията и данните за конфигуриране на контролерите. Използвана е релационна база от данни MySQL 5.3. Посредством SQL заявки контролерите добавят в таблица measures информация за седем машини, свързани към определен контролер. Добавянето на запис, иницииран от контролерите, задейства в базата тригери и съхранени процедури, които оценяват данните и попълват автоматично таблици с информация за работата по смени на отделните машини. Тригерът, задействан при постъпване на данните, актуализира таблица measuresHour. Всеки запис от таблицата съдържа информация за отработеното време за определен час от машините, подключени към контролер с идентификационен номер, записан в полето deviceID. Тригерът вмъква нов запис в началото на всеки час или прави update на полетата от таблицата measuresHour при всяко добавяне на запис в таблица measure. Ето и едно примерно решение на тригера:

```
DELIMITER $$
DROP TRIGGER IF EXISTS measures_INSERT $$
CREATE TRIGGER `measures_INSERT`
AFTER INSERT ON `measures` FOR EACH ROW
BEGIN
 IF minute(new.nTime) = 0 OR ( NOT EXISTS
  ( SELECT 1 FROM measuresHour WHERE
      nDate = new.nDate AND
      nTime = MAKETIME(HOUR(new.nTime),0,0)))
 THEN
    INSERT INTO measuresHour ( deviceID,
       nDate, nTime, Sens0, Sens1)
    VALUES ( new.deviceID, new.nDate,
      MAKETIME( HOUR(new.nTime),0,0),
                new.Sens0, new.Sens1);
    ELSE
      UPDATE measuresHour
      SET
        Sens0 = Sens0 + new.Sens0 ,
        Sens1 = Sens1 + new.Sens1
      WHERE deviceID = new.deviceID AND
        nDate = new.nDate AND
        nTime = MAKETIME(HOUR(new.nTime),0,0);
 END IF;
END $$
DELIMITER;
```

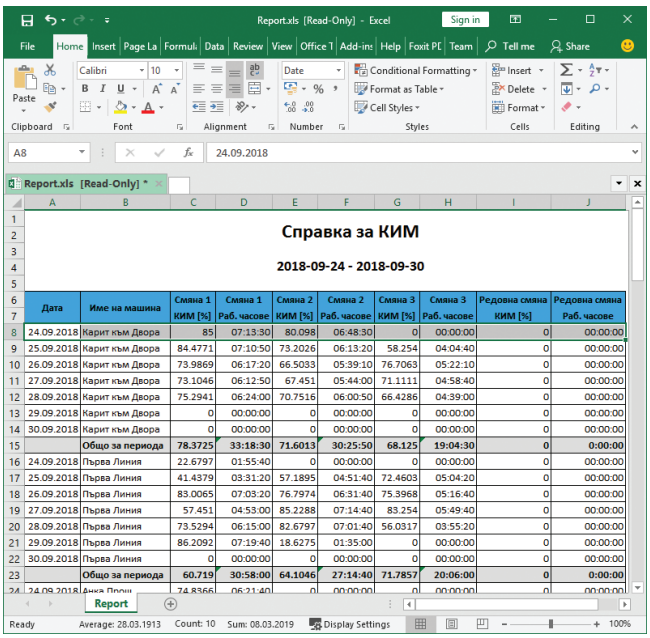
При отсъствие на връзка с даден контролер за период, по-голям от 5 минути, се генерира заявка и се изпраща e-mail до определена група лица, описани в таблица users. Проверката на връзката се извършва от събитие, изпълнявано на всяка минута. Структурата на едно подобно събитие е следната:

```
DELIMITER $$
SET GLOBAL event_scheduler = ON $$
DROP EVENT IF EXISTS eventCheckConnect $$
CREATE EVENT eventCheckConnect
   ON SCHEDULE EVERY 1 minute
   STARTS '2018-04-09 04:30:00'
   DO BEGIN
      CALL checkConnect( );
   END $$
DELIMITER ;
```

Имейлът се генерира от PHP скрипт, задействан от съхранена процедура. Кодът на задействане на PHP скрипта е следният:

```
SET cmd=CONCAT('php.exe -f"/sendMail.php"','');
SET result = sys_eval(cmd);
```

На определен период от време автоматично се генерират дневни, седмични и месечни отчети в Exel формат. Тези отчети се изпращат чрез e-mail към оторизираните лица от PHP скрипт, активиран от базата данни. Месечният отчет се генерира веднъж на първо число, а седмичният – всеки понеделник. На *фиг. 8* е показан един седмичен отчет.



Справка за КИМ

2018-09-24 - 2018-09-30

Дата	Име на машина	Смяна 1 КИМ [%]	Смяна 1 Раб. часове	Смяна 2 КИМ [%]	Смяна 2 Раб. часове	Смяна 3 КИМ [%]	Смяна 3 Раб. часове	Редовна смяна КИМ [%]	Редовна смяна Раб. часове
24.09.2018	Карит към Двора	85	07:13:30	80.098	06:48:30	0	00:00:00	0	00:00:00
25.09.2018	Карит към Двора	84.4771	07:10:50	73.2026	06:13:20	58.254	04:04:40	0	00:00:00
26.09.2018	Карит към Двора	73.9869	06:17:20	66.5033	05:39:10	76.7063	05:22:10	0	00:00:00
27.09.2018	Карит към Двора	73.1046	06:12:50	67.451	05:44:00	71.1111	04:58:40	0	00:00:00
28.09.2018	Карит към Двора	75.2941	06:24:00	70.7516	06:00:50	66.4286	04:39:00	0	00:00:00
29.09.2018	Карит към Двора	0	00:00:00	0	00:00:00	0	00:00:00	0	00:00:00
30.09.2018	Карит към Двора	0	00:00:00	0	00:00:00	0	00:00:00	0	00:00:00
	Общо за периода	78.3725	33:18:30	71.6013	30:25:50	68.125	19:04:30	0	0:00:00
24.09.2018	Първа Линия	22.6797	01:55:40	0	00:00:00	0	00:00:00	0	00:00:00
25.09.2018	Първа Линия	41.4379	03:31:20	57.1895	04:51:40	72.4603	05:04:20	0	00:00:00
26.09.2018	Първа Линия	83.0065	07:03:20	76.7974	06:31:40	75.3968	05:16:40	0	00:00:00
27.09.2018	Първа Линия	57.451	04:53:00	85.2288	07:14:40	83.254	05:49:40	0	00:00:00
28.09.2018	Първа Линия	73.5294	06:15:00	82.6797	07:01:40	56.0317	03:55:20	0	00:00:00
29.09.2018	Първа Линия	86.2092	07:19:40	18.6275	01:35:00	0	00:00:00	0	00:00:00
30.09.2018	Първа Линия	0	00:00:00	0	00:00:00	0	00:00:00	0	00:00:00
	Общо за периода	60.719	30:58:00	64.1046	27:14:40	71.7857	20:06:00	0	0:00:00

Фиг. 8. Седмичен отчет

Отчетът съдържа отработеното време по смени за всеки ден, както и коефициента за използване на машината (КИМ) за всеки ден от седмицата. Накрая се прави обобщение за периода. Коефициентът за използване на машината се изчислява, без да се вземат предвид регулярните почивки. PHP скриптът, отговорен за изпращане на e-mail съобщения с отчетите до оторизираните лица, има приблизително следния код:

```
function sendReportMail() {
 if(!$this->DBLogin()){
  $this->HandleError("DB login failed!");
  return false;
  }
 $sqlMails = "select * from Reports..";
 $retval=$this->connection->query( $sqlMails );
 if($row = $retval->fetch( PDO::FETCH_AS-
 SOC)){
    $mailer = new PHPMailer();
    $mailer->CharSet = 'utf-8';
    $mailer->isSMTP();
    $mailer->SMTPDebug = 0;
    $mailer->Debugoutput = 'html';
```

```
    $mailer->Host = 'smtp.abv.bg';
    $mailer->Port = 465 ;
    $mailer->SMTPSecure = 'ssl';
    $mailer->SMTPAuth = true;
    $mailer->Username = "sys@abv.bg";
    $mailer->Password = "12345";
    $mailer->setFrom('sys@abv.bg');
    $mailer->addReplyTo('sys@abv.bg');
    $mailer->From = 'sys@abv.bg';
    $mailer->Subject = $mSubject;
    $mailer->Body = "Здравейте г-не\n
    На вашето внимание предоставяхме\n"
    .$row['mBody'] ."\n генериран на"
    .$row['added_date']
    ."\n\n С уважение,\n\t\t admin\n ";
    $mailer->addAttachment('Report.xls');
  if(!$mailer->Send()){
    $this->HandleError("Err send email");
    return false;
    }
  unset($mailer);
  return true;
}
```

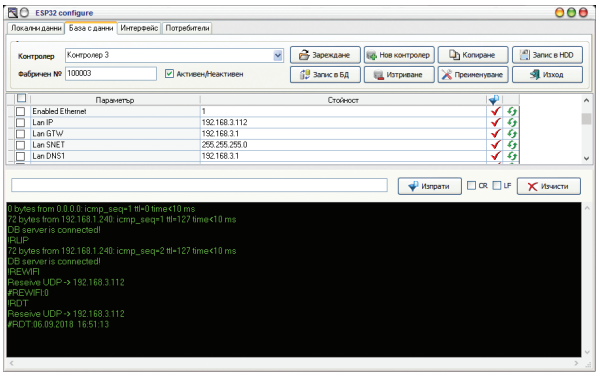
Информацията за отработеното време се съхранява в динамични таблици (view), генерирана от следния скрипт.

```
DELIMITER $$
DROP VIEW IF EXISTS shiftTimeView $$
CREATE VIEW shiftTimeView AS
SELECT hw.deviceID,hw.sensorID, dv.Titles AS
devTitle, sn.Titles AS sensTitle, hw.nDate ,
 SEC_TO_TIME( hw.WorkShift_0 ) AS shiftTime_0,
              hw.WorkShift_0 / 288.0 AS KP0,
 SEC_TO_TIME( hw.WorkShift_1 ) AS shiftTime_1,
              hw.WorkShift_1 / 288.0 AS KP1,
 SEC_TO_TIME( hw.WorkShift_2 ) AS shiftTime_2,
              hw.WorkShift_2 / 288.0 AS KP2,
 SEC_TO_TIME( hw.WorkShift_3 ) AS shiftTime_3,
              hw.WorkShift_3 / 288.0 AS KP3
FROM hoursWorked_new hw, devices dv, sensors sn
WHERE hw.deviceID = dv.deviceID AND dv.Enabled
= TRUE AND sn.deviceID = dv.deviceID AND sn.
sensType > 0 AND hw.sensorID = sn.sensorID;
$$
DELIMITER ;
```

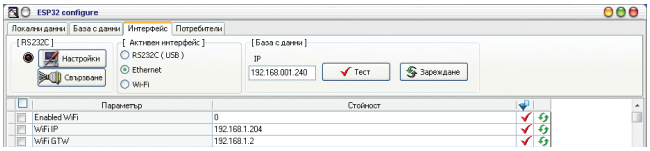
Освен информацията за отработеното време в базата с данни се съхранява и информация за работните графици по машини, информация за настройка на контролерите и сензорите и други. За управлението на тази информация се грижи „Софтуер за конфигуриране и диагностика“ (*фиг. 9*), представляващ третото ниво от системата.

Трето ниво (конфигуриращ и потребителски софтуер)

Целта на софтуера от трето ниво е бързо и лесно въвеждане в експлоатация на системата и генериране на динамични справки. Разработен е в средата C++ Builder 6.0 като 32-битово приложение с възможност за работа под всички версии на Windows след Windows 98. Програмата комуникира с контролера по интерфейсите RS232 и Ethernet, а със сървъра – чрез ODBC драйвери за връзка с MySQL (*фиг. 10*).



Фиг. 9. Софтуер за конфигуриране и диагностика



Фиг. 10. Избор на протокол за връзка с контролерите

Командите за комуникация с контролерите са текстови телеграми със следния формат:

!WIP:192.168.1.152

При правилно приета команда контролера отговаря с телеграма във формат

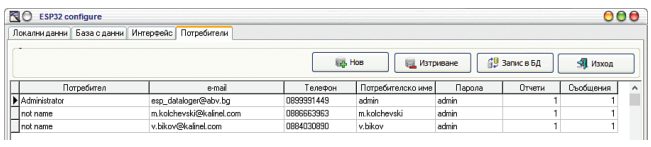
#WIP:192.168.1.152

При възникване на грешка телеграмата за отговор има формат

#WIP:ERROR

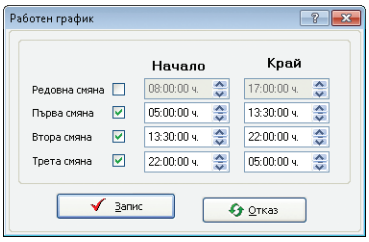
Настройват се следните групи параметри:

- Параметри за връзка по WiFi и Ethernet.
- Параметри за връзка със сървъра.
- Настройка на броя на активните канали.
- Настройване на вида на канала, броячен и по ниво.
- Настройване на работните периоди.
- Сверяване на часовника на контролери (RTC). Ако контролерът е свързан с интернет часовника, се сверява автоматично от сървъра pool.ntp.org.
- Тестване на SD картата.
- Рестартиране на контролер.
- Активиране на процес за четене на данни от SD картата.
- Начална инициализация на контролера.
- Задаване на лица и адреси за получаване на служебни съобщения, отчети и известяване за прекъсване и възстановяване на комуникацията с контролерите (*фиг. 11*) и други.



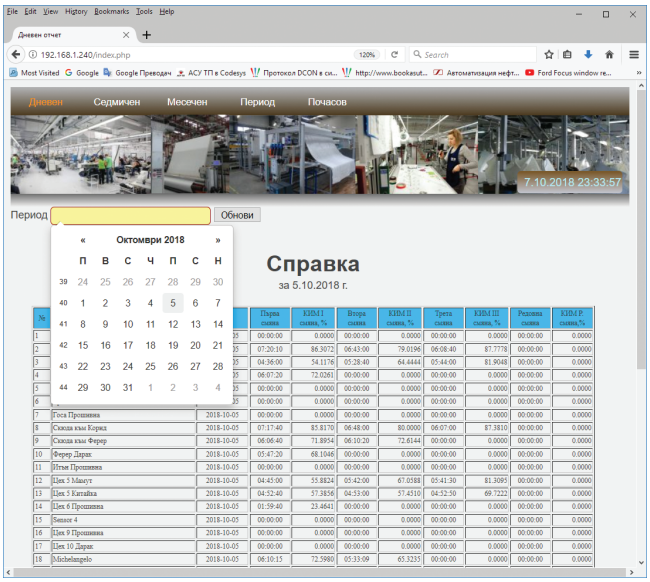
Фиг. 11. Адреси за получаване на съобщения и отчети

Част от изброените параметри се отнасят за конкретната работа на контролерите, друга част, например работните смени (*фиг. 12*), е необходима за сървърното приложение.

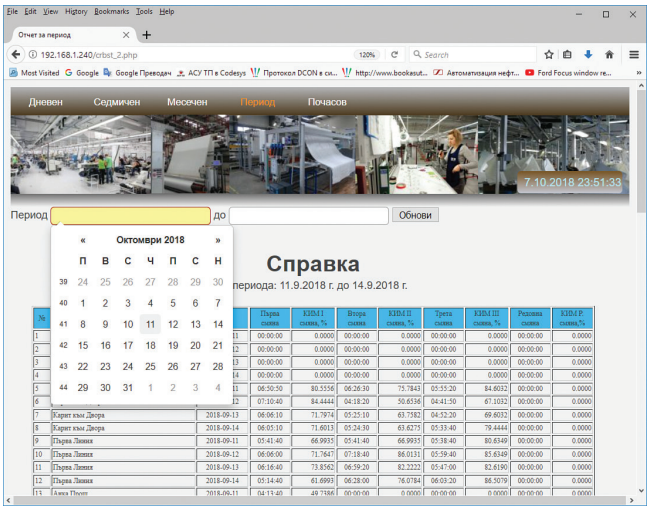


Фиг.12. Работни графици

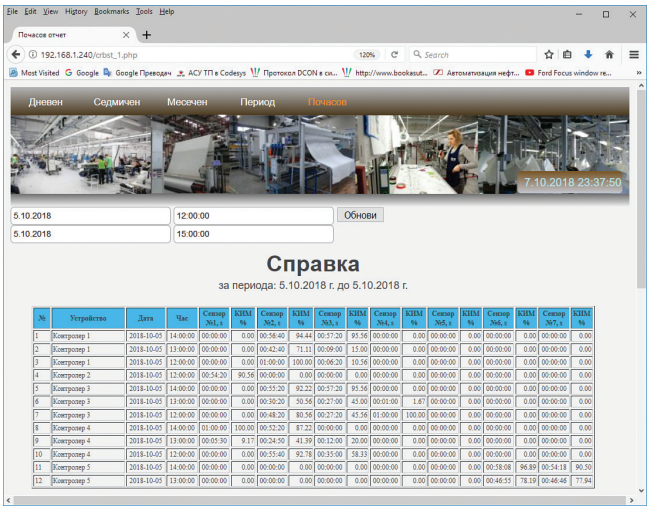
Информацията за настройките може да се съхранява локално на компютъра, от който се извършва настройката или в базата с данни. Позволено е синхронизиране на локалните и сървърните настройки. За улесняване на работата на ръководителите и служителите на фирмата е изградено WEB приложение, позволяващо генериране освен на дневен, седмичен и месечен отчет (фиг. 13) и на справка за период (*фиг. 14*) и почасова справка (*фиг. 15*). Тези две допълнителни справки могат да се използват при оптимизиране на работните графици и разпределение на работниците в смените.



Фиг. 13. Дневен отчет



Фиг. 14. Справка за период



Фиг. 15. Почасова справка

Заключение

Целта на направеното изследване се заключава в решаването на проблеми за разработване на система за автоматизирано отчитане на коефициента на използване на производствените мощности в реално време. Предложеното техническо решение за дистанционен контрол и мониторинг на производствения процес е апробирано и внедрено в текстилното предприятие „Калинел“ ЕООД, Троян и следи работата на 17 текстилни машини, обособени в пет работни групи. Цитираните данни в статията са получени в резултат на реалната експлоатация на системата и се използват за контрол на производствения процес в реално време и за определяне на трудовите възнаграждения на работниците.

Литература

1. Shaul, L., D. Tauber. Critical Success Factors in Enterprise Resource Planning Systems: Review of the Last Decade. – *ACM Computing Surveys,* 45, 2013, 4, 35.

2. Дончев, Д., М. Велев, И. Илиев. Икономика на фирмата. София, Мартинел, 2002.

3. Internet_Things [Electronic Resource]. Available at: https: // en.wikipedia.org/wiki/ Web_web.

4. ESP32 [Electronic Resource]. Available at: https://en.wikipedia.org/wiki/ESP32.

5. Platform with iOs and Android Apps to Control Arduino, ESP8266, Raspberry Pi and Similar Microcontroller Boards over the Internet [Electronic Resource]. Available at: https:// github.com/blynkkk/blynk-server.

6. Internet of Your Things System Blynk [Electronic Resource]. Available at: https://www.blynk.cc/.

7. Vazhdaev, K. V., V. Kh. Abdrakhmanov, R. B. Salikhov. Intellectual System of Residential Zones on the Basis of Information-Measuring Control Systems. – *Electrical and Information Systems and Systems,* 12, 2016, No. 2.

8. Single Board Computer S5P4418 <https://www.graperain.com/ARM-Embedded-S5P4418-Single-Board-Computer/> (09.01.2019).

9. 10 Best Raspberry Pi Alternatives, <https://www.electromaker.io/

blog/article/10-best-raspberry-pi-alternatives> (09.01.2019).

10. NanoPC-T3 Octa-core Cortex A53 Single Board Computer Sells, <https://www.cnx-software.com/2016/04/29/ nanopc-t3-octa-core-cortex-a53-single-board-computer-sells-for-60/> (09.01.2019).

11. Sparky SBC (Single Board Computer), <http://www.robotics4geeks.com/en/arm/4106-sparky-sbc-single-board-computer-0819086010496.html> (09.01.2019).

12. STM32F407IGT6 STM32 Cortex-M4 Development Core Board IO Expander with Onboard NandFlash USB HS/FS Port Ethernet RJ45=XCore407, <https://www.aliexpress.com/ item/STM32F407IGT6-STM32-MCU-core-board-with-IO-expander-2-USB-sockets-Ethernet-NandFlash-Cortex-M4-Development/1286793041.html?spm=a2g0v.10010108.1000001.7.3b487ec7izEST8> (09.01.2019).

13. ESP-WROVER-KIT V3 Getting Started Guide, https://dl.espressif.com/doc/esp-idf/latest/get-started/get-started-wrover-kit.html> (09.01.2019).

14. W5500 Ethernet with POE IOT Board (Arduino Compatible), https://www.dfrobot.com/product-1286.html

15. WiDo - An Arduino Compatible IoT (internet of thing) Board, <https://www.dfrobot.com/product-1159.html> (09.01.2019).

16. ESP32 WROOM Series, <https://www.espressif.com/en/products/hardware/esp-wroom-32/overview> (09.01.2019).

17. ESP32-GATEWAY development board with WiFi BLE Ethernet, micro SD card UEXT and GPIO, <https://www.olimex.com/Products/IoT/ESP32/ESP32-GATEWAY/open-source-hardware > (09.01.2019).

За контакти:
Доц. д-р **Станимир Йорданов**
Ас. д-р **Георги Михалев**
Катедра „Автоматика, информационна и управляваща техника“
Технически университет – Габрово
e-mails: sjjordanov@mail.bg
gimihalev@mail.bg