Tímový projekt ZS 2017

Dokumentácia pre teplotnú testovaciu komoru

Vypracoval	Rafael Gajanec
Dátum	28. septembra 2019
Verzia dokumentu	1.0

1. Základné časti riadiacej dosky

STM32F205VGT6	32-bitový mikrokontrolér
M24C16-FDW6TP	I2C 16 kB EEPROM
FN1-1001W4SM00BW	Primárny biely 7-segmentový displej
–	Obvod pre ovládanie primárneho 7-segmentového displeja pomocou GPIO
OPD-S3913LY-BW	Sekundárny žltý 7-segmentový displej
OPD-S3913UPG-BW	Sekundárny zelený 7-segmentový displej
MAX6954ATL	SPI ovládač sekundárnych 7-segmentových displejov
62AG11-L5-020C	Kvadratúrny panelový enkodér
–	Panelové tlačidlá pripojené ku GPIO
CP2102N-A01-GQFN24	Prevodník USB/UART
ADM3251EARWZ	Izolovaný prevodník RS232/UART
MAX31865ATP	SPI R/T prevodník pre meranie teploty pomocou RTD
–	Obvod analógového prúdového zdroja, riadený výstupom z DAC
SHT31-DIS-B	I2C digitálny senzor okolitej teploty a vlhkosti
VEML7700	I2C digitálny senzor osvetlenia
CMI-1240-SMT	Akustický 4 kHz indikátor
–	Obvody na ovládanie vzduchovej pumpy a ventilátorov pomocou GPIO
–	Interné regulátory napätia

2. Priradenie vstupov a výstupov STM32

Pin	Signál	Typ	Popis
PA0	FC4	DO	Riadenie ventilátora 4
PA1	FF4	DI	Spätná väzba ventilátora 4
PA2	FC5	DO	Riadenie ventilátora 5
PA3	FF5	DI	Spätná väzba ventilátora 5
PA4	AOUTB	AO	Výstup B z DAC
PA5	AOUTA	AO	Výstup A z DAC
PA6	ALRT0	DI	Prerušenie zo senzora teploty a vlhkosti na doske (v miestnosti)
PA7	ALRT1	DI	Prerušenie z externého senzora teploty a vlhkosti (v komore)
PA8	HC7	DO	Invertovaný výstup pre ovládanie výkonového mostíka B (J7-1)
PA9	TX1	DO	Výstup sériových dát do prevodníka USB/UART
PA10	RX1	DI	Vstup sériových dát z prevodníka USB/UART
PA11	CTS1	DO	Výstup pre kontrolu toku dát z prevodníka USB/UART
PA12	RTS1	DI	Vstup pre kontrolu toku dát do prevodníka USB/UART
PA13	SWDIO	DIO	Vstup/výstup pre programovanie cez SWD
PA14	SWCLK	DI	Hodinový vstup pre programovanie cez SWD
PA15	WHBK	DO	Výstup pre nastavovanie jasu bieleho displeja
PB0	HC2	DO	Invertovaný výstup pre ovládanie výkonového mostíka A (J6-3)
PB1	HC3	DO	Invertovaný výstup pre ovládanie výkonového mostíka A (J6-1)
PB2	NC		Nepripojený pin
PB3	SCLK1	DO	Hodinový výstup pre SPI pripojené k obom R/T prevodníkom
PB4	MISO1	DI	Dátový vstup pre SPI pripojené k obom R/T prevodníkom

Pin	Signál	Typ	Popis
PB5	MOSI1	DO	Dátový výstup pre SPI pripojené k obom R/T prevodníkom
PB6	SDA1	DIO	Obojsmerné sériové dáta pre senzor osvetlenia, senzory teploty a vlhkosti a pre EEPROM
PB7	SCL1	DO	Hodinový výstup pre senzor osvetlenia, senzory teploty a vlhkosti a pre EEPROM
PB8	FC7	DO	Riadenie ventilátora 7
PB9	FF7	DI	Spätná väzba ventilátora 7
PB10	ENCA	DI	Vstup A z kvadratúrneho enkodéra
PB11	ENCB	DI	Vstup B z kvadratúrneho enkodéra
PB12	DDCS	DO	Selektovací výstup pre SPI pripojené ku ovládaču sekundárnych 7-segmentových displejov
PB13	SCLK2	DO	Hodinový výstup pre SPI pripojené ku ovládaču sekundárnych 7-segmentových displejov
PB14	MISO2	DI	Dátový vstup pre SPI pripojené ku ovládaču sekundárnych 7-segmentových displejov
PB15	MOSI2	DO	Dátový výstup pre SPI pripojené ku ovládaču sekundárnych 7-segmentových displejov
PC0	LED1	DO	Panelová LED, môže byť použitá pre indikáciu znižovania teploty
PC1	LED0	DO	Panelová LED, môže byť použitá pre indikáciu zvyšovania teploty
PC2	LED2	DO	Panelová LED, môže byť použitá pre indikáciu toku dát cez RS232/UART
PC3	LED3	DO	Panelová LED, môže byť použitá pre indikáciu toku dát cez USB/UART
PC4	HC0	DO	Invertovaný výstup pre ovládanie výkonového mostíka A (J6-2)
PC5	HC1	DO	Invertovaný výstup pre ovládanie výkonového mostíka A (J6-4)
PC6	FC0	DO	Riadenie ventilátora 0
PC7	FF0	DI	Spätná väzba ventilátora 0
PC8	FC1	DO	Riadenie ventilátora 0
PC9	FF1	DI	Spätná väzba ventilátora 0
PC10	HC6	DO	Invertovaný výstup pre ovládanie výkonového mostíka B (J7-3)
PC11	HC5	DO	Invertovaný výstup pre ovládanie výkonového mostíka B (J7-4)
PC12	HC4	DO	Invertovaný výstup pre ovládanie výkonového mostíka B (J7-2)
PC13	WCN	DO	Výstup pre riadenie zápisu do EEPROM
PC14		XI	Vstup pre 32.768 kHz signál z kryštálu
PC15		XO	Budiaci výstup pre 32.768 kHz kryštál
PD0	SW0	DI	Vstup z pravého krajného panelového spínača
PD1	SW1	DI	Vstup z pravého stredného panelového spínača
PD2	SW2	DI	Vstup z ľavého stredného panelového spínača
PD3	SW3	DI	Vstup z ľavého krajného panelového spínača
PD4	DRDY0	DI	Vstup z prvého R/T prevodníka indikujúci pripravenosť dát na čítanie
PD5	MCS0	DO	Selektovací výstup pre SPI pripojené k prvému R/T prevodníku
PD6	DRDY1	DI	Vstup z druhého R/T prevodníka indikujúci pripravenosť dát na čítanie
PD7	MCS1	DO	Selektovací výstup pre SPI pripojené k druhému R/T prevodníku
PD8	TX3	DO	Výstup sériových dát do prevodníka RS232/UART
PD9	RX3	DI	Vstup sériových dát z prevodníka RS232/UART
PD10	APD	DO	Výstup pre zapnutie/vypnutie vzduchovej pumpy
PD11	BUZ	DO	Výstup pre zapnutie/vypnutie piezoelektrického akustického indikátora
PD12	FC2	DO	Riadenie ventilátora 2
PD13	FF2	DI	Spätná väzba ventilátora 2
PD14	FC3	DO	Riadenie ventilátora 3
PD15	FF3	DI	Spätná väzba ventilátora 3
PE0	WH3	DO	Výstup pre riadenie spoločnej katódy bieleho 7-segmentového displeja 0
PE1	WH4	DO	Výstup pre riadenie spoločnej katódy bieleho 7-segmentového displeja 1

Pin	Signál	Typ	Popis
PE2	WH0	DO	Výstup pre riadenie spoločnej katódy bieleho 7-segmentového displeja 2
PE3	WH2	DO	Výstup pre riadenie spoločnej katódy bieleho 7-segmentového displeja 3
PE4	WH1	DO	Výstup pre riadenie spoločnej katódy bieleho 7-segmentového displeja 4
PE5	FC6	DO	Riadenie ventilátora 6
PE6	FF6	DI	Spätná väzba ventilátora 6
PE7	WHA	DO	Výstup pre riadenie segmentov A
PE8	WHB	DO	Výstup pre riadenie segmentov B
PE9	WHC	DO	Výstup pre riadenie segmentov C
PE10	WHD	DO	Výstup pre riadenie segmentov D
PE11	WHE	DO	Výstup pre riadenie segmentov E
PE12	WHF	DO	Výstup pre riadenie segmentov F
PE13	WHG	DO	Výstup pre riadenie segmentov G
PE14	WHDP	DO	Výstup pre riadenie segmentov DP
PE15	SWENC	DI	Vstup zo spínača integrovaného v kvadratúrnom enkodéri
PH0		XI	Vstup pre 16 MHz signál z kryštálu
PH1		XO	Budiaci výstup pre 16 MHz kryštál

Typy signálov sú nasledovné:
DO – digitálny výstup
DI – digitálny vstup
DIO – digitálny obojsmerný signál
AO – analógový výstup
XI – vstup z kryštálového filtra
XO – budiaci výstup pre kryštálový filter

3. Elektronika mimo riadiacej dosky

3.1. Výkonový mostík

...

3.2. Lineárny regulátor

...

3.3. Zásuvný riadiaci modul pre sieťové zdroje

...

4. Program pre STM32

Súčasná verzia kódu je napísaná v jazyku C s využitím funkcií operačného systému ChibiOS 17.6 a príslušných knižníc. Dôvodom pre použitie operačného systému bolo výrazné zjednodušenie implementovania funkcií, ktoré potrebujú byť vykonávané paralelne v reálnom čase, pričom požiadavky na presnosť časovania sú nízke, až žiadne.

Digitálna kópia programovacieho prostredia je prílohou tohto dokumentu. Archív je potrebné rozbaliť priamo na disk <C:\> a následne prostredie možno spustiť otvorením súboru <C:\ChibiStudio\start_gcc63.bat>. Projektové súbory sú uložené v priečinku <C:\ChibiStudio - STP\chibios176\demos\STM32\RT-STM32F207-DISCOVERY\>.

4.1. Polia premenných

V prvej časti kódu sa nachádzajú 3 polia statických globálnych integerov. Tieto sú určené na zápis a čítanie zvonku pre vzdialené ovládanie teplotnej komory.

Prvé pole `variableArea0` je určené pre riadenie a PID reguláciu teploty, tieto hodnoty majú byť pre čítanie aj zápis. Pole `variableArea1` je určené iba pre čítanie hodnôt z rôznych senzorov a prevodníkov. Tretie pole `variableArea2` je určené pre parametre používateľskej konfigurácie.

Doteraz vyhradené adresy v uvedených poliach sú popísané priamo v kóde a v kóde je tiež legenda, podľa ktorej je možné interpretovať jednotlivé hodnoty, ako napríklad rôzne módy, pozícia desatinnej čiarky a podobne. V prípade potreby je možné veľkosť polí ďalej rozšíriť o nové premenné.

4.2. Funkcie a vlákna

```
void toggleLed(short n)
```

Vstupom tejto funkcie je číslo od 0 do 3, zodpovedajúce štyrom LED na paneli. Po zavolaní funkcie sa zmení stav vybranej LED na opačný.

```
void setLed(short n)
```

Vstupom tejto funkcie je číslo od 0 do 3, zodpovedajúce štyrom LED na paneli. Po zavolaní funkcie sa rozsvieti vybraná LED.

```
void clearLed(short n)
```

Vstupom tejto funkcie je číslo od 0 do 3, zodpovedajúce štyrom LED na paneli. Po zavolaní funkcie sa vypne vybraná LED.

```
static THD_FUNCTION(thdBlinkLed0, arg)
```

Po spustení tohto vlákna sa LED0 rozbliká s periódou rovnou hodnote priradenej premennej `blinkLed0` v milisekundách. Toto vlákno je dobré najmä pre demonštráciu a oboznámenie sa s formátom deklarácie vláken pre ChibiOS a ich volaním. Pod definíciou vlákna je uvedený krátky ukážkový kód, ktorý je možné skopírovať do `int main(void)`.

```
static THD_FUNCTION(thdBlinkLed1, arg)
```

Pozri `static THD_FUNCTION(thdBlinkLed0, arg)`.

```
static THD_FUNCTION(thdBlinkLed2, arg)
```

Pozri `static THD_FUNCTION(thdBlinkLed0, arg)`.

```
static THD_FUNCTION(thdBlinkLed3, arg)
```

Pozri `static THD_FUNCTION(thdBlinkLed0, arg)`.

```
int readKey(short n)
```

Vstupom tejto funkcie je číslo od 0 do 3, zodpovedajúce štyrom tlačidlám na paneli. Výstupom funkcie je 1 ak je vybrané tlačidlo stlačené, inak je výstupom 0. Pri nesprávnom volaní funkcie je výstupom -1.

```
static THD_FUNCTION(thdEncoder, arg)
```

Treba napísať kód, ktorý bude spracúvať údaje z kvadratúrneho enkodéra na paneli. Je možné napríklad vytvoriť nové vlákno, alebo použiť prerušenia. Pôvodne špecifikovaný enkóder mal disponovať vstavaným tlačidlom, doručený však bol nesprávny typ.

Jedným z možných riešení pre spracovanie signálov z enkodéra môže byť ICU.

```
static THD_FUNCTION(thdCom, arg)
```

Toto vlákno obsluhuje komunikáciu prostredníctvom zbernice RS232. Jeho úlohou je prijímať príkazy z počítača a vhodným spôsobom odpovedať. Komunikačný protokol je inšpirovaný protokolom Omron CompoWay/F, ktorý je dobre spracovaný v dokumente *H175 – Digital Temperature Controllers – Communications Manual*.

Vlákno má za úlohu analyzovať prijatý reťazec ASCII znakov, identifikovať chyby ak sa nejaké vyskytnú pri prenose a spracovať príkazy a dáta. Zároveň má slúžiť na čítanie a zápis dát vo vyššie spomenutých poliach globálnych premenných.

Kód tohto vlákna je neúplný. Je potrebné dopísať kód pre identifikovanie niektorých možných chýb. Bolo by vhodné, aby bolo číslo uzlu nastaviteľné pomocou zatiaľ nešpecifikovanej premennej v poli `variableArea2`, ktoré je momentálne naprogramované na hodnotu "99". Najdôležitejšie je však doprogramovať spracovanie príkazov a dát, napríklad podľa dokumentu *H175 – Digital Temperature Controllers – Communications Manual*.

```
static THD_FUNCTION(thdWhiteBacklight, arg)
```

Toto vlákno reguluje jas bieleho displeja na základe hodnôt v poliach globálnych premenných. Je potrebné napísať kód pre čítanie údajov zo senzora osvetlenia a na základe toho regulovať jas automaticky.

```
int binPwr(short x)
```

Výstupom tejto funkcie je x-tá mocnina čísla 2.

```
void whiteChar(char ascii, short dp)
```

Vstupom tejto funkcie je jeden znak a indikátor desatinnej čiarky. Na základe toho sa aktivujú vhodné segmenty bieleho displeja.

```
void whiteDig(short n)
```

Vstupom tejto funkcie je číslo, na základe ktorého sa aktivujú zodpovedajúce cifry na bielom segmentovom displeji.

```
void whiteClr(void)
```

Táto funkcia deaktivuje biely displej.

```
static THD_FUNCTION(thdWhiteDisplay, arg)
```

Vlákno zobrazuje na bielom displeji znaky, ktoré sú uložené v premennej `whiteData`.

```
static THD_FUNCTION(thdBlinkWhiteDigit, arg)
```

Program v tomto vlákne rozbliká jednu cifru na bielom displeji na základe hodnoty priradenej v premennej `blinkWhiteDigit`. Perióda blikania je rovná dvojnásobku hodnoty `DIG_BLINK_PERIOD` v milisekundách. Toto vlákno má význam do budúcna pre ovládanie prostredníctvom panelových tlačidiel a enkodéra.

```
void regFansCtrl(short enable)
```

Ak je táto funkcia zavolaná s hodnotou 0, tak sa vypnú oba ventilátory na chladenie regulátorov, inak sa aktivujú.

```
void extFansCtrl(short enable)
```

Ak je táto funkcia zavolaná s hodnotou 0, tak sa vypnú všetky štyri vonkajšie ventilátory na komore, inak sa aktivujú.

```
void intFansCtrl(short enable)
```

Ak je táto funkcia zavolaná s hodnotou 0, tak sa vypnú oba vnútorné ventilátory, inak sa aktivujú. Tieto ventilátory nie sú momentálne nainštalované.

```
void regFansUpdate(int pwm)
```

Funkcia zabezpečuje, že PWM signál nebude generovaný za nevhodných podmienok, ktoré sú ohraničené zdola `MIN_CB_PWM` a zhora `MAX_CB_PWM`. Nízka hodnota by totiž zapríčinila nepravidelné pulzy, pri vysokej hodnote zas nemá zmysel generovať pulzy.

```
void extFansUpdate(int pwm)
```

Pozri `void regFansUpdate(int pwm)`.

```
static void pwm3pcb(PWMDriver *pwmp)
```

Funkcia volaná generátorom PWM signálu na začiatku každej periódy.

```
static void pwm3c1cb(PWMDriver *pwmp)
```

Funkcia volaná generátorom PWM signálu po dosiahnutí požadovanej šírky pulzu.

```
static void pwm3c2cb(PWMDriver *pwmp)
```

Funkcia volaná generátorom PWM signálu po dosiahnutí požadovanej šírky pulzu.

```
static THD_FUNCTION(thdFanControl, arg)
```

Využitím vyššie opísaných funkcií, toto vlákno ovláda ventilátory regulátorov a vonkajšie a vnútorné ventilátory komory na základe preddefinovaných hodnôt a hodnôt v poli `variableArea2`. Kód umožňuje jednoduchú automatickú reguláciu otáčok ventilátorov, ale v takom prípade môže byť potrebné doladiť preddefinované parametre.

```
int readFanFeedback(short n)
```

Vstupom tejto funkcie je číslo od 0 do 7, zodpovedajúce ôsmim ventilátorom. Výstupom je stav signálu spätnej väzby.

```
int countToRpm(int count)
```

Funkcia prepočítava hodnotu v počítadle pulzov na otáčky za minútu.

```
int rpmToCount(int rpm)
```

Funkcia prepočítava otáčky za minútu na hodnotu počítadla pulzov.

```
static THD_FUNCTION(thdFanFeedback, arg)
```

V tomto vlákne sa vyhodnocuje spätná väzba z ventilátorov a zapisuje sa na príslušné adresy v poli `variableArea2`.

```
static void dacOutInit(void)
```

Funkcia aktivuje DAC a nastaví oba výstupy na nulovú hodnotu.

```
static int rtdReadRegister8(unsigned char addr, unsigned char rtd)
```

Funkcia prečíta 8-bitové dáta zvoleného prevodníka MAX31856 z registra podľa zadanej adresy.

```
static int rtdReadRegister16(unsigned char addr, unsigned char rtd)
```

Funkcia prečíta 16-bitové dáta zvoleného prevodníka MAX31856 z registra podľa zadanej adresy.

```
static void rtdWriteRegister8(unsigned char addr, unsigned char data, unsigned char rtd)
```

Funkcia zapíše 8-bitové dáta do registra zvoleného prevodníka MAX31856 na zadanej adrese.

```
static void rtdReadFault(unsigned char rtd)
```

Funkcia prečíta chybový register zvoleného prevodníka MAX31856 a zapíše ich na adresu príslušnej premennej v poli `variableArea1`.

```
static void rtdClearFault(unsigned char rtd)
```

Po zavolaní funkcia vyčistí chybový register zvoleného prevodníka MAX31856.

```
static void rtdEnableBias(unsigned char rtd)
```

Funkcia aktivuje bias pre senzor pripojený k zvolenému prevodníku MAX31856. Aktivácia je nevyhnutná pre meranie teploty, avšak vo veľmi špecifických prípadoch môže spôsobený samoohrev spôsobovať problémy pri meraní a vtedy je vhodné bias deaktivovať po odobratí vzorky.

```
static void rtdDisableBias(unsigned char rtd)
```

Funkcia deaktivuje bias pre senzor pripojený k zvolenému prevodníku MAX31856.

```
static void rtdEnableAuto(unsigned char rtd)
```

Funkcia aktivuje automatickú konverziu pre zvolený prevodník MAX31856.

```
static void rtdDisableAuto(unsigned char rtd)
```

Funkcia deaktivuje automatickú konverziu pre zvolený prevodník MAX31856.

```
static int rtdReadValue(unsigned char rtd)
```

Funkcia prečíta dáta zo zvoleného prevodníka MAX31856, zodpovedajúce pomeru referenčného odporu a odporu platinového snímača teploty.

```
static double rtdReadTemp(double rnom, double rref, unsigned char rtd)
```

Funkcia prepočíta dáta zo zvoleného prevodníka MAX31856 na teplotu v stupňoch Celzia.

```
static void rtdInit(void)
```

Túto funkciu je potrebné zavolať na počiatočné nastavenie oboch prevodníkov MAX31856. Pod touto funkciou je

```
static void fetCtrl(unsigned char fet, unsigned char mode)
```

Vstupom tejto funkcie je číslo `fet` od 0 do 7, identifikujúce jeden z ôsmich tranzistorov a číslo `mode`, nastavením ktorého na hodnotu rovnú 1 sa aktivuje vybraný tranzistor, inak sa tranzistor otvorí.

```
static void bridgeCtrl(unsigned char mode)
```

Vstupom tejto funkcie je číslo `mode` od 0 do 3, ktorým sa súbežne nastavujú oba tranzistorové mostíky do rôznych konfigurácií zapojení. Tranzistory sa pred prepnutím do novej konfigurácie vždy otvoria po dobu definovanú v premennej `HC_TSW`, čo slúži ako prevencia pred skratovaním napájania termoelektrických modulov.

```
void helloWorld(void)
```

Funkcia rozbliká panelové LED po stlačení jednotlivých tlačidiel.

```
void mainInit(void)
```

```
void mainExit(void)
```

```
int main(void)
```