

Instrukcja programowania płytek edycji 2014

Spis treści

| 1. Opis płytki procesorowej (sumo_base_5) | 1 |
|---|----|
| 1.1. Podstawowe elementy płytki | 2 |
| 1.2. Pozostałe elementy płytki | 3 |
| 2. Opis płytki sterującej napędami (sumo_motor_5) | 3 |
| 2.1. Elementy płytki | 4 |
| 2.2 Parametry układu | 4 |
| 3. Zalecane rozmieszczenie czujników i silników | 5 |
| 4. Programowanie układu przez USB (DfuSe) krok po kroku | 5 |
| 5. Konfiguracja środowiska CooCox i kompilacja programu | 8 |
| 5.1. Instalacja i przygotowanie środowiska | 8 |
| 5.2. Otwieranie i kompilacja projektu | 9 |
| 5.3. Przygotowanie pliku *dfu | 10 |
| 6. Debugowanie za pomocą programatora ST-link | 11 |
| 6.1. Przygotowanie środowiska | 11 |
| 6.2. Debugowanie | 12 |
| 6.3. Powrót do pracy z bootloaderem | 13 |
| | |

1. Opis płytki procesorowej (sumo_base_5)

Układ stanowi "mózg" robota, posiada złącza czujników, interfejsy komunikacyjne, diody LED, przyciski i przełączniki ogólnego przeznaczenia. Jest on oparty na mikroprocesorze STM32F103RBT6 z rodziny ARM Cortex-M3. Układ można programować na trzy sposoby:

- Prze interfejsu USB, za pomocą zaprogramowanego bootloadera DFU (Device Firmware Upgrade). **Metoda zalecana**, opisana w punkcie 4.
- Przez interfejs SWD, za pomocą programatora ST-link. Metoda pozwala na debugowanie programu.
- Przez interfejs USART (port szeregowy).

1.1. Podstawowe elementy płytki



- LED1 11 diody LED do dyspozycji użytkownika (programisty)
- LED12 dioda LED informująca o włączeniu portu USB
- LED_PWR zielona dioda, świeci gdy zasilanie jest podłączone
- SWITCH1 3 przełączniki do dyspozycji programisty
- SWITCH4 Wybór trybu bootowania. Poza programowaniem przez UART, ZAWSZE W POZYCJI ON
- BUTTON1, 2 przyciski do dyspozycji użytkownika (programisty)
- RESET Przycisk resetu
- GRD1 5 wejścia czujników podłoża (do wykrywania krawędzi ringu)
- DIST1 4 wejścia czujników odległości (do wykrywania przeciwnika)

1.2. Pozostałe elementy płytki



- USB służy głównie do programowania procesora. Może być też wykorzystany w innych celach
- SWD Złącze debuggera.
- USART port szeregowy, pozwala na komunikację z procesorem w czasie działania programu. Może też służyć do programoania.
- EXT złącze z wyprowadzonymi niewykorzystanymi sygnałami procesora. Zawiera 4 kanały przetwornika ADC, interfejsy SPI i I2C. Umożliwia np. podłączenie dodatkowych czujników odległości, karty SD, akcelerometru i żyroskopu.

2. Opis płytki sterującej napędami (sumo_motor_5)

Układ odpowiedzialny jest za zasilanie płytki bazowej i serwomechanizmów, oraz sterowanie silnikami DC. Za wytwarzanie napięcia 5V odpowiedzialna jest przetwornica impulsowa ST1S10.

2.1. Elementy płytki



- Zasilanie złącze baterii, zakres napięć zasilania: 6V 14V. Zalecane jest stosowanie baterii litowo-polimerowych 7,2V lub 11,1V. UWAGA NA PRAWIDŁOWE PODŁACZENIE BATERII ('+' baterii do '+' płytki, '-' baterii do '-' płytki)
- MOTOR1, 2 złącza silników DC (prądu stałego, szczotkowych)
- SERVO1, 2 złącza serwomechanizmów

2.2 Parametry układu

- Zakres napięć zasilania: **6-14V**.
- Maksymalny prąd pobierany przez serwa i płytka procesorowa: 1,5A
- Maksymalny prąd pobieramy przez każdy z silników: 4A
- Wymiary: **55x55mm**

Po złożeniu zestaw powinien wyglądać następująco:



3. Zalecane rozmieszczenie czujników i silników

Na schemacie pokazano zalecane rozmieszczenie czujników odległości (DIST), podłoża (GRD) i silników (MOTOR). Takie rozmieszczenie obsługuje program podstawowy.



4. Programowanie układu przez USB (DfuSe) krok po kroku

Metoda pozwala na wgrywanie plików *.dfu (sposób ich przygotowania opisany w dalszej części instrukcji). Niezbędne do tego jest pobranie DfuSe.zip, rozpakowanie i zainstalowanie programu. Program można też pobrać ze strony:

http://www.st.com/web/catalog/tools/FM147/CL1794/SC961/SS1533/PF257916

1. Uruchomić aplikację DfuSe Demo.

| 🗢 DfuSe Demo (v3.0.3) |
|---|
| Available DFU Devices Available DFU Devices Application Mode: Vendor ID: Vendor ID: Vendor ID: Procuct ID: Procuct ID: Vendor ID: |
| Select Target(s): Target Id Name Available Sectors (Double Click for more) |
| Upload Action File: Choose Upload Transferred data size 0 KB(0 Bytes) of 0 KB(0 Bytes) Operation duration 00:00:00 Upgrade of Verify Action File: Verify Action File: Fi |
| Abort Quit |

2. Kliknąć na przycisk *Choose* i wybrać plik *.dfu.

| 🧼 DfuSe Demo (| v3.0.3) | | | | | |
|---|-----------------------------|---|---------------|---------------|------------|------------------|
| Available DFU De | evices | | - Application | Modo: | | |
| Otwieranie | | | | | | ? 🛛 |
| Szukaj w: Moje bieżące dokumenty Pulpit Moje dokumenty Moje dokumenty Moje komputer | Pulpit | :y eciowe) | • | ← 10 <u>c</u> | * ==- | |
| Moje miejsca sieciowe | Nazwa pliku: Pliki typu: | minisumo.dfu Dfu Files (*.dfu) Otwórz tylko do od | lozytu | | • [• _ | Otwórz Anuluj |
| Abort | | | | | | Quit |

- 3. Podłączyć płytkę do portu USB, płytka musi być zasilana np. z baterii.
- 4. Trzymając wciśnięty przycisk BUTTON1 na płytce wcisnąć RESET. Spowoduje to uruchomienie bootloadera.



- 5. Powinna się zapalić dioda LED12 i mrugać dioda LED1
- 6. Po kilku sekundach w polu *Available DFU Devices* powinno się pojawić *STM Device in DFU Mode*.

| 🧼 DfuSe Demo (v | 3.0.3) |
|----------------------|--|
| - Aveilable DFU Devi | ices |
| STM Device in DF | U Mode DFU Mode: DFU Mode: |
| Suprem 11-1- | ed Mapifeetati wurstant Vendor ID: Vendor ID: 0483 |
| Supports Down | nload Accelerated Upload (ST) Procuct ID: Procuct ID: DF11 |
| 🗹 Can Detach | Version: Version: 0200 |
| Enter DFU mode/ | HID detach Leave DFU mode |
| Actions | |
| Select Target(s): | Target Id Name Available Sectors (Double Click for more) |
| | 00 Internal Flash 128 sectors |
| | 01 SPI Flash : M25P64 128 sectors |
| | |
| | |
| Upload Action | Upgrade or Verify Action |
| File: | File: minisumo.dtu |
| Channe | Vendor ID: U483 11 argets in the |
| | Procuct ID: 0000 |
| Transferred data : | size Version: 0000 |
| 0 KB(0 Bytes) of (| 0 KB(0 Bytes) |
| | Butiniae Upstalia duration (Bemove some FFs) |
| Operation duration | n |
| 0 | .0:00:00 Upgrade Verity |
| | |
| | File correctly loaded. |
| Abort | Quit |

- 7. Zaznaczyć opcję Verify after download i kliknąć Upgrade.
- 8. Przy pytaniu *Continue however* (wyskakujące okno) kliknąć *Tak*.

| 🧼 DfuSe Demo (v3.0.3) |
|---|
| Available DFU Devices |
| STM Device in DFU Mode |
| ☑ Supports Upload ☐ Manifestation tolerant ☑ Supports Download ☐ Accelerated Upload (ST) ☑ Can Detach |
| Actions Select Target(s): Target Id Name Available Sectors (Double Dick for more) DfuSeDemo |
| Your device was plugged in DFU mode. So it is impossible to make sure this file is correct for this device. |
| Upload Action Continue however ? |
| File: Tak Nie |
| Choose Procuct ID: 0000 |
| Version: 0000 |
| Operation duration Operation duration |
| 00:00:00 Choose Upgrade Verify |
| File correctly loaded. |
| Abort Quit |

- 9. Jeśli wszystko pujdzie dobrze, na dolnym pasku postępu powinien pojawić się napis V*erify successful* !
- 10. Aby wystartować wgrany program wcisnąć przycisk RESET na płytce,.

5. Konfiguracja środowiska CooCox i kompilacja programu

Środowisko CooCox CoIDE i zestaw narzędzi GCC umożliwiają napisanie w języku C i kompilację programu dla robota.

5.1. Instalacja i przygotowanie środowiska

- 1. Pobrać środowisko CoIDE ze strony: <u>http://www.coocox.org/CooCox_CoIDE.htm</u>
- 2. Pobrać zestaw narzędzi GCC ze strony: <u>https://launchpad.net/gcc-arm-embedded/</u> +download
- 3. Zainstalować CoIDE i GCC, zapamiętać, gdzie został zainstalowany GCC
- 4. Uruchomić CoIDE, wybrać *Project* → *Select Toolchain Path*

| 🚺 Select Toolchain Path | |
|--|--------|
| Toolchain Path: (the directory where "arm-none-eabi-gcc.exe"existed) | Browse |
| | |
| | ОК |

5. W oknie dialogowym wygrać ścieżkę do GCC (wcześniej zapamiętaną)

5.2. Otwieranie i kompilacja projektu

- 1. Pobrać plik minisumo_coocox.zip bibliotekę i przykładowy program dla robota
- 2. Rozpakować go w katalogu workspace programu CoIDE.
- 3. Uruchomić CoIDE.
- 4. Kliknąć Project → Open project i wybrać plik minicumo.coproj z rozpakowanego katalogu.



5. Aby edytować program, otworzyć w środowisku plik main.c



- 6. Kliknąć na przycisk build na górnym pasku
- 7. W oknie w dolnej części ekranu pojawi się log z kompilacji i ewentualne komunikaty o błedach
- 8. Aby program wgrać na płytkę, trzeba przygotować plik *.dfu (opisane poniżej), lub posłużyć się programatorem ST-link (opisane w punkcie 6)

5.3. Przygotowanie pliku *dfu

Konwersja do plików *dfu jest potrzebna, jeśli ma zostać użyty bootloader opisany w punkcie 4.

- 1. Uruchomić program *DFU File Manager* (zainstalowany wraz z DfuSe Demo).
- 2. Wybrać opcję pierwszą



3. Kliknąć na *S19 or HEX*.

| 🛱 DFU File Manager (v3.0 | .3) - Generation |
|--|--|
| Vendor ID 0x 0493 Product ID 0x 0000 Version 0x 0000 | Injection Target ID: 0 S19 or Hex Multi BIN Target Name: ST Deletion Delete selected Image |

4. Wybrać plik minisumo.hex. Znajduje się on w podkatalogu /minisumo/Debug/bin katalogu z projektem minisumo (z pubktu 5.2.). Być może trzeba będzie zmienić typ pliku na *.hex w oknie wyboru pliku.

| Otwieranie | | | | | ? 🛛 |
|---|-----------------------------|--|---|---------|------------------|
| Szukaj w: | in 🔁 bin | | • | 🗢 🗈 💣 🎫 | - |
| Moje bieżące dokumenty Pulpit Moje dokumenty Moje dokumenty | minisumo.hex | | | | |
| Moje miejsca sieciowe | Nazwa pliku: Pliki typu: | minisumo.hex hex Files (".hex) Otwórz tylko do odczytu | | • | Otwórz Anuluj |

- 5. Kliknąć na Generate i podać nazwę i wybrać miejsce zapisu pliku *.dfu.
- 6. Dalsze postępowanie opisane w punkcie 4.

6. Debugowanie za pomocą programatora ST-link

Programator ST-link umożliwia debigowanie kodu programu. Ay taki posiąść, najtaniej jest zaopatrzyć się z którąś z płytek ewaluacyjnych STM32xDISCOVERY, np. STM32F0DISCOVERY. Użycie programatora powoduje nadpisanie bootloadera, jednak nic nie stoi na przeszkodzie, żeby do niego później wrócić.

6.1. Przygotowanie środowiska

- 1. Pobrać i zainstalować sterownik do ST-linka: http://www.st.com/web/catalog/tools/FM146/CL1984/SC724/SS1677/PF251168
- 2. Wykonać czynności z punktów 5.1 i 5.2.
- 3. Ustawić zworki na płytce DISCOVERY w pozycji rozwartej
- 4. Połączyć złącza SWD płytki DISCOVERY i procesorowej, zwracając uwagę, gdzie jest pierwszy pin. Kolejność pinów płytki procesorowej jest taka sama jak dla układu STM32F0DISCOVERY.



- 5. Włączyć programator do portu USB komputera.
- 6. W środowisku CoIDE kliknąć na *View* \rightarrow *Configuration*.
- 7. Przejść do karty Link i wartość *IROM1* ustawić z 0x08003000 na 0x08000000.

| CooCox CoIDE C:/CooCox/CoIDE/workspace/ | ninisumo/minisumo.coproj | |
|---|---|------------------|
| File Edit View Project Flash Debug Search Help | | |
| i 📫 🗎 : 🛗 🏯 🍫 🛎 🖋 i 💿 💿 ji | 合 : 毘 ⑧ : 〃 ・ : 如 - 전 - や 수 ・ 수 · | |
| Components 🛛 🗖 🗖 | le main.c lo Configuration ⊠ | " 🗖 🕼 Help 🛛 📃 🗖 |
| Composition is as Composition is as Composition is a Composition is Composition is Composition i | Configuration evente Comple Link or put User Debugger Download ind pervice Comple Link or put User Debugger Download ind pervice Comple Link or put User Debugger Download ind pervice Comple Link or put User Debugger Download ind pervice Comple Link or put User Debugger Download ind pervice Comple Link or put User Debugger Download ind pervice Comple Link or put User Debugger Download ind pervice Comple Link or put User Debugger Download ind pervice Comple Link or put User Debugger Download ind pervice Comple Link or put User Debugger Download ind pervice Comple Link or put User Debugger Download ind pervice Comple Link or put User Debugger Download ind pervice Comple Link or put User Debugger Download pervice Comple Comple Comple Link or put User Debugger Download pervice Comple | |
| 0x00000000 in ?? () kill | | |
| < | | |
| i 😭 minisumo | ji 👌 Sign in | n to CooCox |

- 8. W pliku sumoconf.h zmienić _BOOTLOADER_PRESENT z 1 na 0
- 9. Skompilować projekt ponownie.

6.2. Debugowanie

1. Kliknąć na ikonę Start Debug

| CooCox ColDE C:/CooCox/ColDE/workspace/ | ninisumo/minisumo.coproj | |
|---|--|-------------------|
| File Edit View Project Flat Debug Search Help | | |
| i 📫 🗎 i 🛗 👪 🚺 🔯 🎽 🖉 i 😳 🎯 i | 🔗 : 🗒 🚳 : 😕 🛷 - : : 🍠 : 🛃 - 🖓 - 약 수 - 수 - | |
| Components 🛛 🖓 🖓 | 🖻 main.c 🛛 | |
| Device [STM32F103R8] | <pre>S5// wattosc aktuanie wykonywanel czynnosci S6 volatile int currentAction = STOP; S7 S8 int main(void) (9 init(); 60 // dawanie znaku zycia 61 rprint("ResetVrlm"); 62 // uruchomienie timera 63 timerSchedule(timer1, 1); 64 65 // oczekiwanie na nacisniecie przycosku 1 odliczanie 66 countdown(); 67 67 68 // petla glowna programu 69 for (;;) (70 // wykon wici 71 selectAction(); 72 // wykonywanie akcji 64 65 // petla glowna programu 66 // petla glowna programu 66 // petla glowna programu 67 70 // wykonywanie akcji 67</pre> | |
| E Console | | |
| <terminated> minisumo.configuration [Cortex-M Application]</terminated> | :\Program Files\GNU Tools ARM Embedded\4.7 2012q4\bin\arm-none-eabi-gdb (01-03-2014 11:43) | |
| Program received signal SIGINT, Inter 0x00000000 in ?? () kill | rupt. | |
| i 🖆 minisumo i 🗗 | Writable Smart Insert 1:1 | Sign in to CooCox |

- 2. Środowisko zmieni perspektywę
- 3. Aby uruchomić aplikację kliknąć Run, możliwe też jest ustawianie pułapek, podglądanie zmiennych bądź praca krok po kroku

| CooCox CoIDE C:/CooCox/CoIDE/workspace | /minisumo/minisumo.coproj | |
|--|--|---------------------|
| File Edit View Debug Search Help | | |
| i i i i 🌔 🕨 🕕 🖬 🖓 🖓 🖓 🖓 | - 1 🗾 1 🖗 - 🖗 - 🏷 - | |
| Project 🖂 📄 🖶 🗸 🖓 | 🖻 main.c 🖾 💽 startup_stm32f10x_md.c | Disassembly 🛛 🗖 🗖 |
| Target minisumo | <pre>55// wartosc aktuanie wykonywanej czynnosci 56 volatile int currentAction = STOP; 57 58 int main(void) (59 init(); 60 // dawanie rnaku rycia 61 rprintf("ResetUrin"); 62 // uruchomienie timera 63 timerSchedule(timeri, 1); 64 65 // oczekiwanie na nacisniecie przycosku i odliczanie 66 countdown(); 67 68 // petla glowna programu 69 for (;;) (70 // wykor akcji 71 selectAction(); 72 // wykonywanie akcji </pre> | Enter location here |
| Eonsole 🅸 Debug | 🙀 🧔 🏹 🗖 🗖 🗠 Semihosting 😒 🕅 Variables | 🔓 🚮 👒 🗖 🗖 |
| minisumo.configuration [Cortex: M Application] GDB Hardware Debugger (01-03-2014 11:44) (Susj Thread [1] (Suspended: Breakpoint hit.) Thread [1] (Suspended: Breakpoint hit.) Thread [1] (Suspended: Breakpoint hit.) Tamain() main.ctS9 0x800104cl - C1/Program Files(GAU Tools ARM Embedded[4.7 2C - C1(CooCox(CoIDE)workspace minisumo minisum0[0 | ended) 12q4[bin]am-none-eabi-gdb (01-03-2014 11:44) ebug[bin]minisumo.eff (01-03-2014 11:44) | 2 8 |
| 🔁 🚔 minisumo. | Writable Smart Insert 59 : 1 😤 S | iign in to CooCox |

6.3. Powrót do pracy z bootloaderem

- 1. Pobrać i zainstalawać program CoFlash: http://www.coocox.org/CoFlash_Programmer.htm
- 2. Pobrać plik DFU.elf
- 3. Uruchomić program CoFlash
- 4. W zakładce *Config* wybrać układ STM32F103RB

| 🕞 CooCox Cortex Flash Program | ner 📃 🗆 🔀 |
|--|---|
| Config Command | |
| STM32F101VG STM32F101ZC STM32F101ZD STM32F101ZE STM32F102C4 STM32F102C6 STM32F102C8 STM32F102C8 STM32F102R4 STM32F102R6 STM32F102R8 STM32F102R8 STM32F102R8 STM32F102R8 STM32F102R8 STM32F102R8 STM32F102R8 STM32F102R8 STM32F103C8 | USB Adapter Setup Adapter: ST-Link Adapter: ST-Link Adapter: ST-Link Adapter: SWD Adapter: SWD Adapter: SYSRESETREQ Address Trace Microcontroller Info Vendor: ST RAM Start: 0x20000000 Plash Driver Description Size Address Range STM32F10x Med-densit 128K 0800000H - 0801FFFFH |
| STMS2F103RB | Add Remove |
| | Close |

5. Przejść do zakładki *Command*, w polu *Data File* wskazać plik DFU.elf

| 🕫 CooCox Cortex Flash Programmer | | | | | |
|-----------------------------------|-------------|---------------|-------|----------|--|
| Config Command | | | | | |
| Flash Sector Map | | | | | Due en en |
| STM32F10x Med-density 128KB Flash | | | | | Program Data St |
| | | | | – | |
| | Sector | Start Address | Size | ~ | C:\Documents and Settings\Administrator\Pulj |
| E | Z 0 | 0×08000000 | 0x400 | | 🔽 Auto Frase |
| E | 2 1 | 0×08000400 | 0x400 | | Program With: 🔽 Auto Verify |
| E | 2 2 | 0×08000800 | 0x400 | | Run After Program |
| E | 23 | 0×08000C00 | 0x400 | | Sector Officety Duo |
| E | 24 | 0×08001000 | 0x400 | | Sector Onset: UXU |
| E | 2 5 | 0×08001400 | 0x400 | | |
| E | 26 | 0×08001800 | 0x400 | | Operate |
| E | 27 | 0×08001C00 | 0x400 | | Program |
| E | 28 | 0×08002000 | 0x400 | | |
| E | 29 | 0×08002400 | 0x400 | | |
| E | 2 10 | 0×08002800 | 0x400 | | |
| E | 2 11 | 0×08002C00 | 0x400 | | Useful Photochast |
| E | 12 | 0×08003000 | 0x400 | | |
| E | 13 | 0×08003400 | 0x400 | | |
| E | Z 14 | 0×08003800 | 0x400 | ~ | |
| , | | | | | |
| | | | | | |
| | | | | | Close |
| | | | | | |

6. Kliknąć *Program*. Po zaprogramowaniu procesor przejdzie w tryb DFU.