ROBOTI

VE ŠKOLE PRO PRAKTICKOU VÝUKU, MOTIVACI I ZÁBAVU

CZ.1.07/1.1.24/01.0066

ROBOTI

Ing. Michal ŘEPKA, Ph.D.

Střední škola elektrotechnická, Ostrava, Na Jízdárně 30, příspěvková organizace

2014

esf evropský sociální fond v ČR

EVROPSKÁ UNIE

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

OP Vzdělávání pro konkurenceschopnost

INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

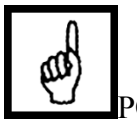
POKYNY KE STUDIU:

 ČAS KE STUDIU

Čas potřebný k prostudování látky. Čas je pouze orientační a slouží jako hrubé vodítko pro rozvržení studia kapitoly.

 CÍL

Cíle, kterých lze dosáhnout prostudováním kapitoly – konkrétní dovednosti, znalosti.

 POJMY K ZAPAMATOVÁNÍ

Pojmy, které si je potřeba zapamatovat.

 VÝKLAD

Teoretický výklad studované látky, zavedení nových pojmů a jejich vysvětlení.

 ŘEŠENÉ PŘÍKLADY

Podrobný postup při řešení příkladů.

 SHRNUTÍ POJMŮ

Zopakování hlavních pojmů.

 OTÁZKY

Několik teoretických otázek pro ověření zvládnutí kapitoly.

 PRAKTICKÉ ÚLOHY

Několik praktických příkladů pro ověření zvládnutí kapitoly.

OBSAH:

1. Úvod


ČAS KE STUDIU: 10 minut


CÍL

Prostudováním této kapitoly získá čtenář přehled o tom co je myšleno složitějším robotem.


VÝKLAD

V předchozím dokumentu – Manipulátory a vozíky – byl robotický systém Bioloid využíván pouze ke konstrukci robotů s maximálně sedmi pohonnými jednotkami. Díky otevřenosti a modularitě jednotlivých komponent, je možné tento systém používat také, ke konstrukci složitějších robotů, tj. robotů, které svým tvarem reprezentují nějakého živého tvora. V tomto dokumentu je možné najít návod na sestavení robota typu Humanoid, Pes a Dinosaurus.

Každý takový robot je tvořen minimálně 15-ti pohony a jedním nebo několika senzory, sloužící k lepší interaktivitě s okolním prostředím. Pokud vznikne požadavek vytvořit nějaký pohyb takovýmto robotem, je nutné ovlivňovat, někdy i současně, všechny pohonné jednotky, ze kterých je robot sestaven. A právě tento specifický požadavek dělá robota složitějším.

Další problém nastává v situaci, kdy chceme provést nějaký pohyb, který je z hlediska konstrukce robota možný, ale dílčí výkon konkrétního pohonu to neumožňuje. Proto daný pohyb musí být realizován s ohledem na výkon jednotlivých pohonů a uživatel musí pečlivě dbát na základní fyzikální Newtonowy zákony a skládání sil. Pokud je toto přehlíženo, tak obvykle dochází k přetěžování některých pohonů, což vede k jejich zničení.

V neposlední řádě, u takovýchto robotů, bývá problém zajistit stabilitu, tj. aby robot při chůzi nebo pohybu jednoduše nepadal. Aby tento požadavek byl dodržen, je potřeba vytvořit hlubší analýzu robota a požadovaného pohybu. Tato analýza obvykle vyžaduje znalost vyšší matematiky, což začátečníky může často odradit.

2. Software RoboPlus Motion

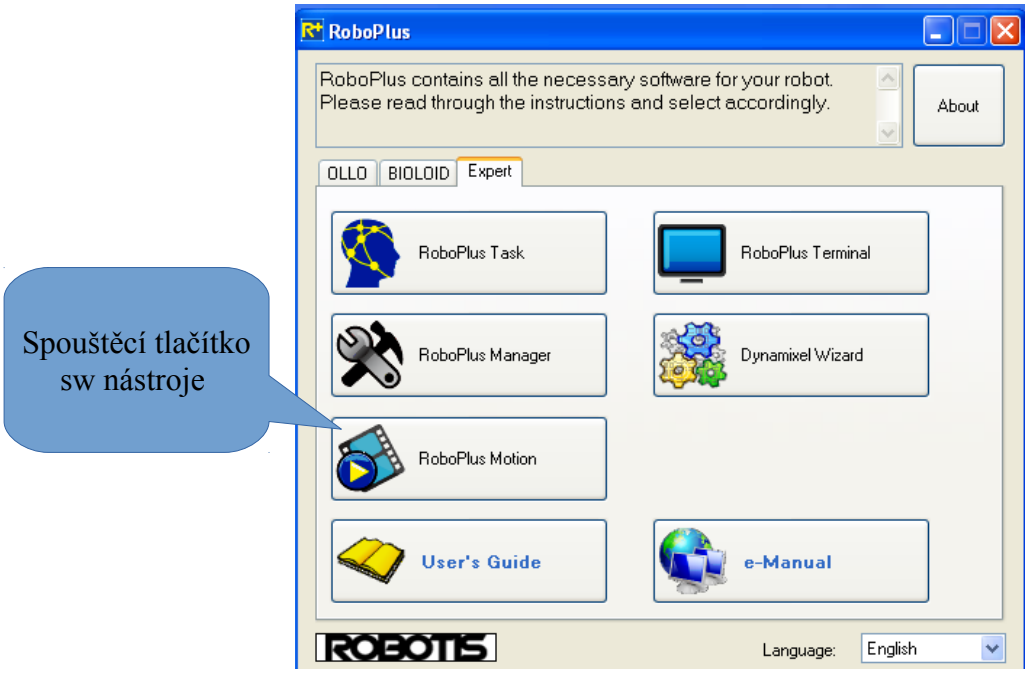
ČAS KE STUDIU: 45 minut

CÍL

Prostudováním této kapitoly získá čtenář základní představu, jak využít software RoboPlus Motion pro přípravu a realizaci pozic a pohybů připojeného robota. Po vyzkoušení tohoto nástroje čtenář zjistí, že tento nástroj tuto práci velmi usnadňuje.

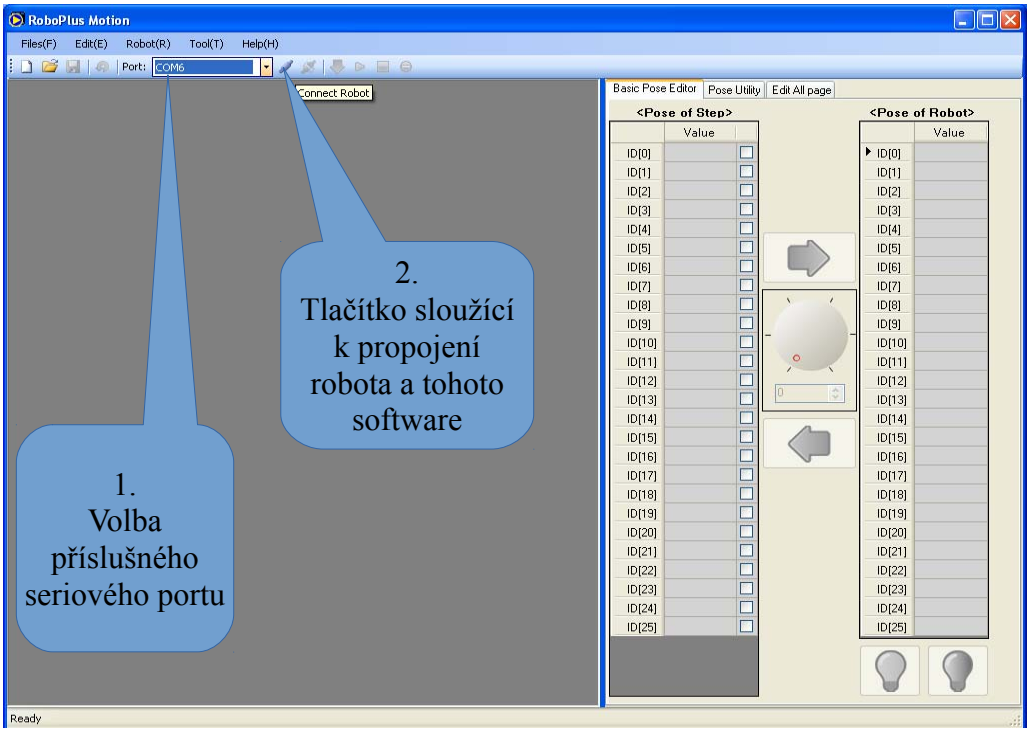
VÝKLAD

V předchozím dokumentu byl představen programovací nástroj s názvem RoboPlus Task. Tento nástroj je možné používat také pro přípravu a realizaci pozic a pohybů složitějších robotů. Avšak použitím softwarového nástroje RoboPlus Motion je tato činnost mnoho-násobně urychlena.



Obr. 2.1 – Hlavní obrazovka software RoboPlus Studia

Abychom mohli vytvořit návrh nějakého pohybu, je potřeba, abychom měli k dispozici robota a toho připojili k počítači. Poté, v programu, stačí zvolit příslušný seriový port a zmáčknout tlačítko „Connect“. Tímto dojde k načtení vnitřní paměti řídicí jednotky do počítače a uživatel může začít editovat jednotlivé pohybové skupiny dat.



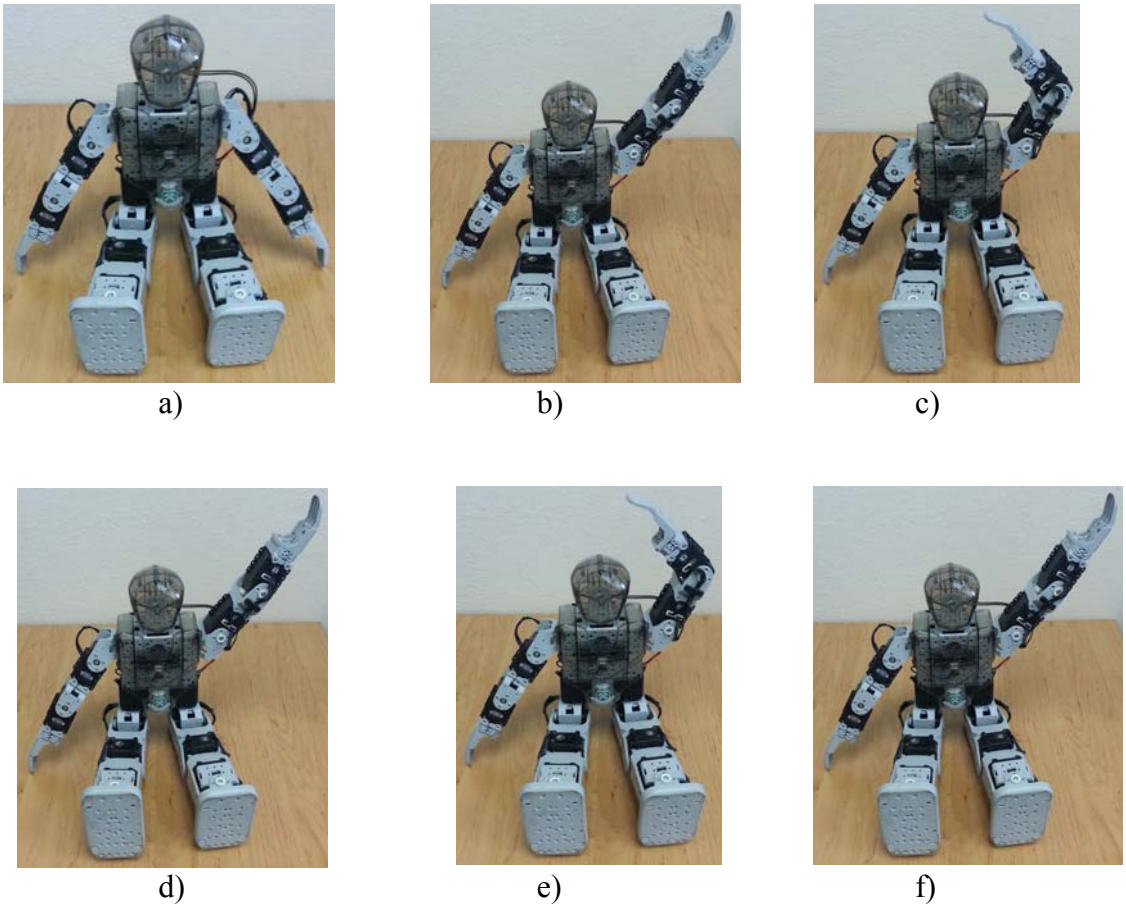
Obr. 2.2 – Hlavní obrazovka software RoboPlus Motion

Výsledná data je možné zálohovat i v počítači a kdykoliv později je možné tato data editovat, případně je nahrát do řídicí jednotky. Pokud pohybová data (tzv. Soubor s příponou mtn) jsou zapsána pouze v řídicí jednotce a vznikne potřeba je editovat, pak vystačí připojit robota a data se automaticky načtou tím, že se robot propojí se softwarem. Poté je možné tato data zálohovat, ve formě souboru s mtn příponou, kdekoliv na disk.

Toto je základní rozdíl mezi programem RoboPlus Task, kde jednou zapsaný program v řídicí jednotce už nelze editovat, pokud jej nemáme někde zálohovaný.

2.1. Postup vytváření pohybových dat

Předpokládejme, že budeme pracovat s robotem typu Humanoid, u kterého bychom chtěli vytvořit pohyb rukou v sedící poloze. Každý pohyb, který chceme realizovat, si musíme na počátku rozdělit na jednotlivé pozice, které pak budeme postupně vykonávat. Podobně jak se tvoří animovaný film. Viz. následující obrázek.



a) b) c)

d) e) f)

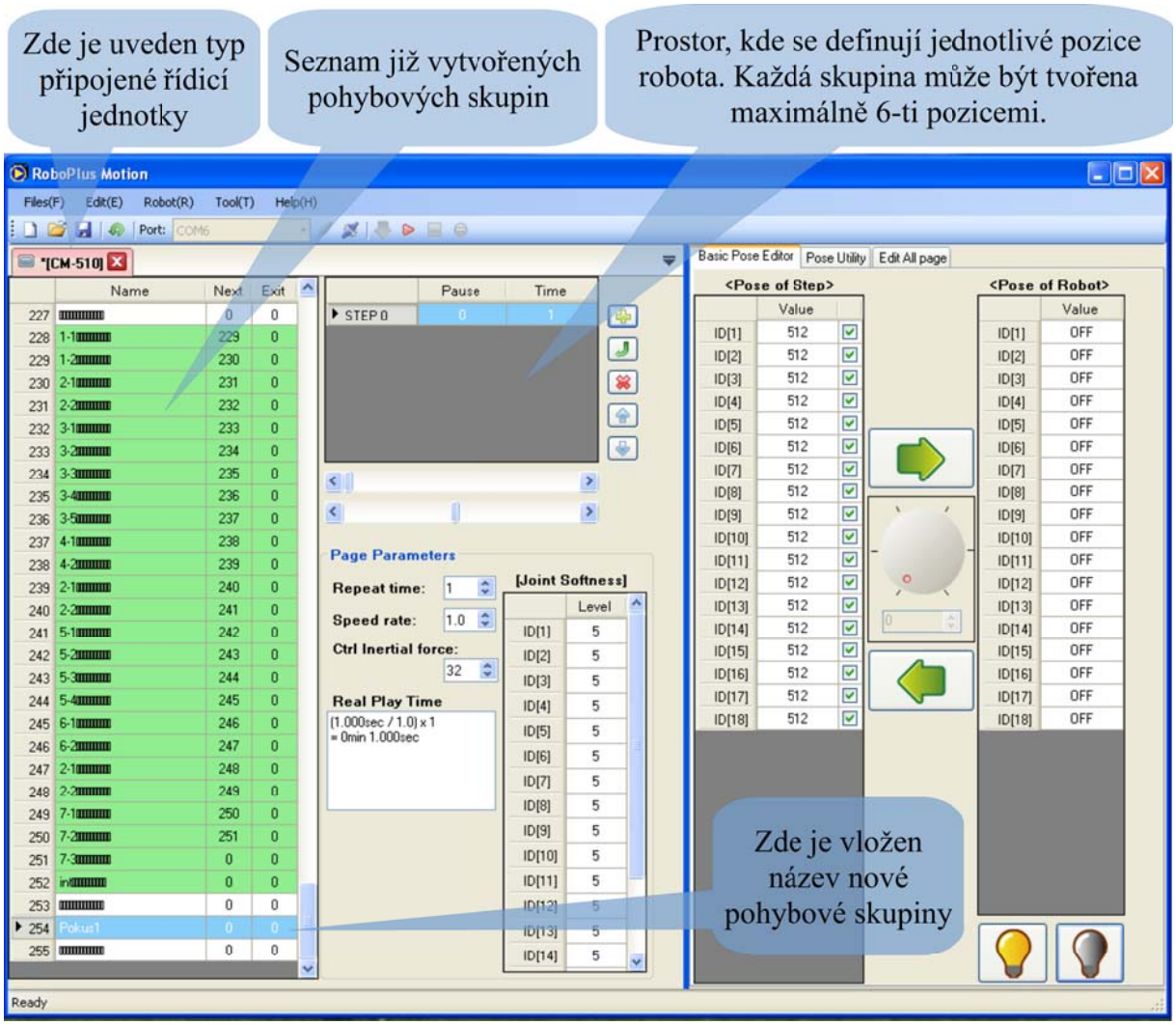
Obr. 2.3 – Navržené jednotlivé pozice robota - mávání

U každé nové skupiny pozic, jenž budou tvořit nějaký pohyb, je potřeba zvolit počáteční, neboli výchozí pozici. Jedná se o pozici, o které předpokládáme, že se v ní nachází robot, když spustíme vykonávání jednotlivých pozic.

Např. kdybychom v tomto případě spustili podprogram vykonávání tohoto pohybu paží (mávání), když robot stojí, tak při provádění první pozice by pravděpodobně spadl do nedefinované polohy, protože pro tuto skupinu pozic je výchozí pozice – sedící robot.

Nyní bude následovat postup definování nové skupiny pozic, které po spuštění mohou vytvářet komplikovaný pohyb. V tomto případě to bude pohyb připomínající mávání rukou, Tak jak je rozdělen na pozice na obrázku 2.3.

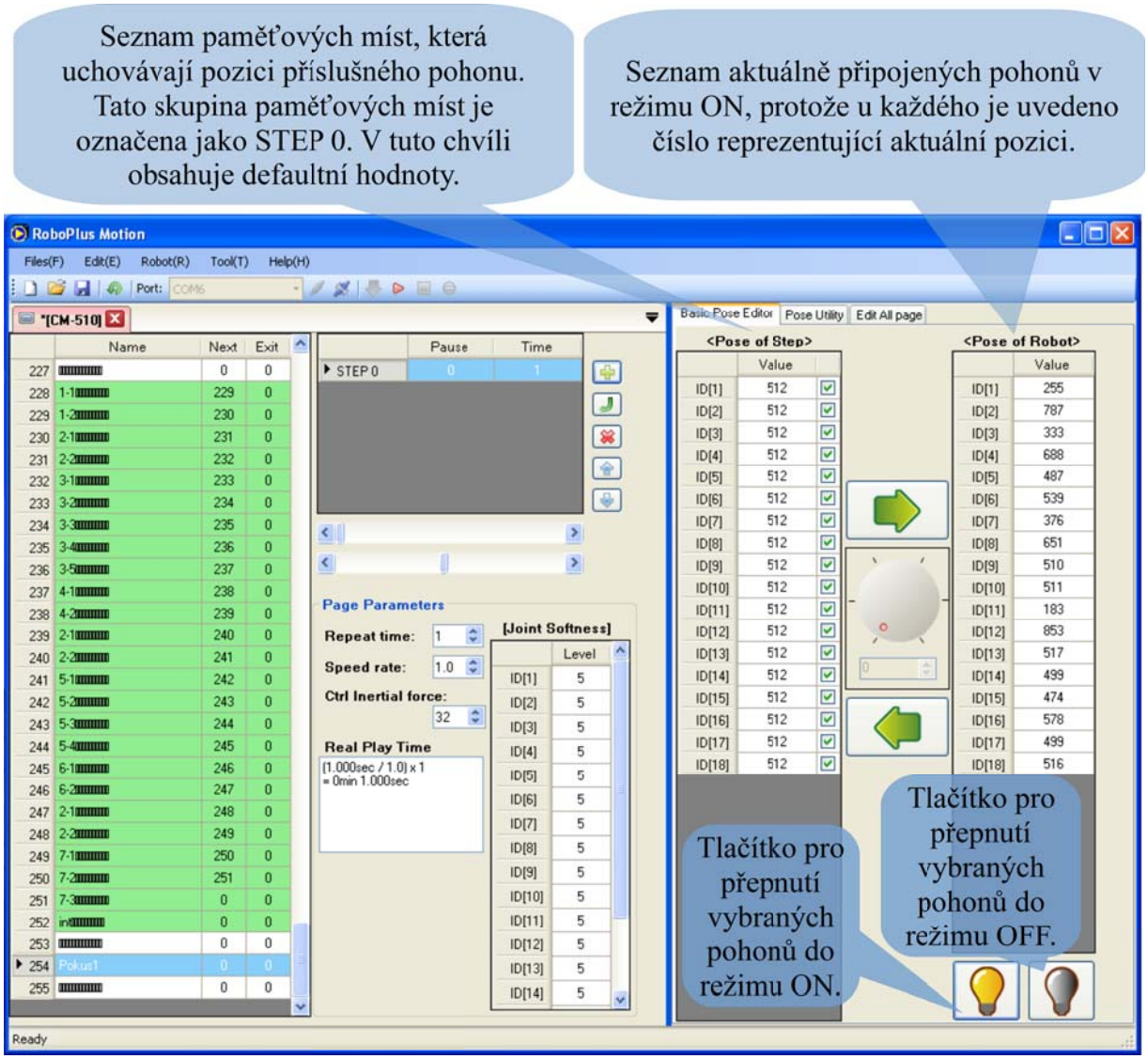
Po připojení se v levé části zobrazí již nadefinované pohybové skupiny pozic a nyní je potřeba vybrat volné paměťové místo a tam vložit název pro novou skupinu pozic.



Obr. 2.4 – Definování nové pohybové skupiny s názvem Pokus 1

Po propojení robota s tímto software jsou všechny pohony ve stavu OFF, což znamená, že připojeny jsou pouze v módu, kdy dochází pouze ke čtení jejich pozic. Tento stav lze vidět v pravé části obrazovky, která je označena <Pose of Robots>. Pokud je v tomto sloupci uvedeno slovo OFF (např. Obr 2.4), znamená to, že příslušný pohon má vypnutý motor a uživatel s tímto pohonem může ručně manipulovat.

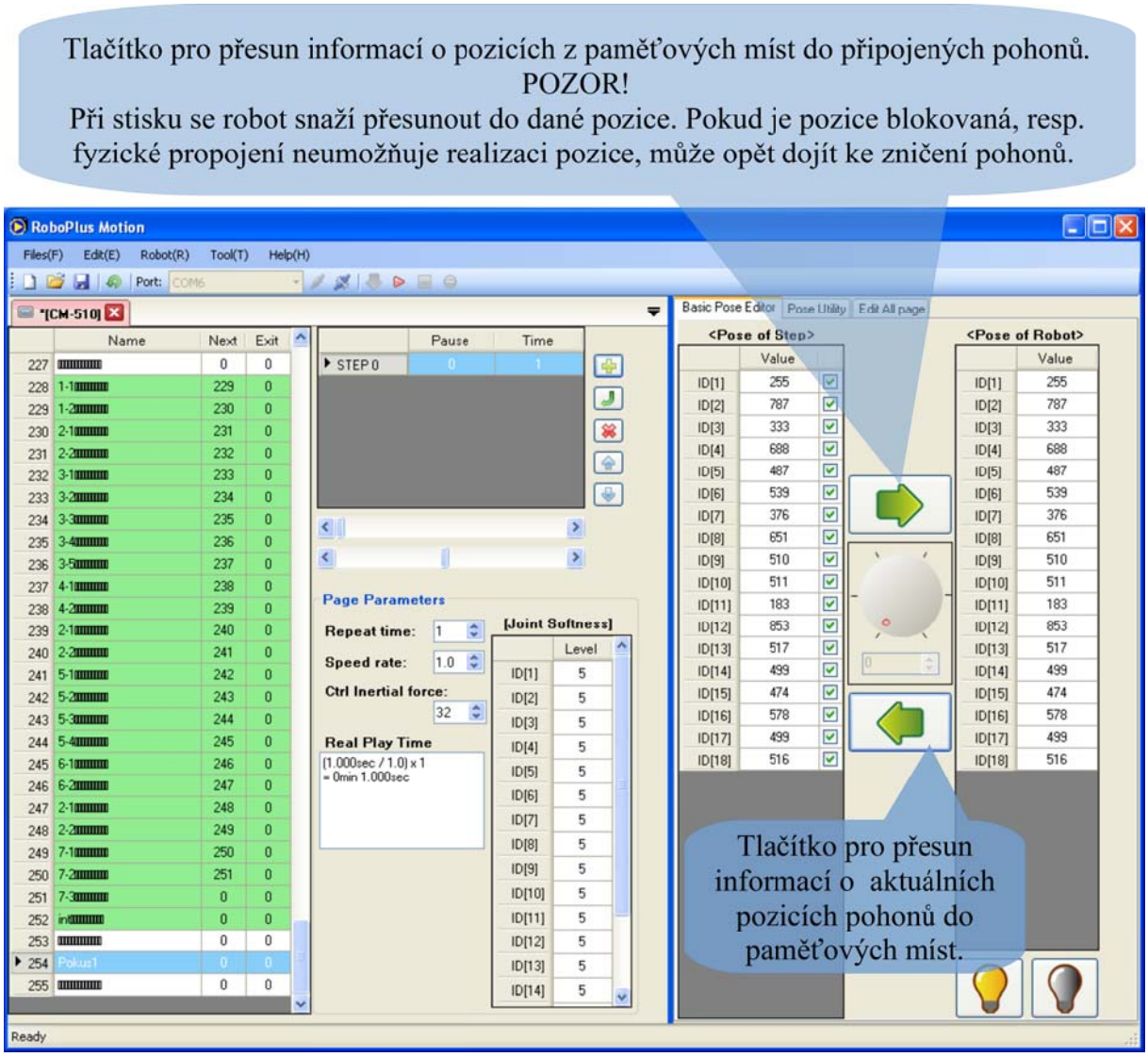
V okamžiku, kdy si uživatel takovýmto způsobem vytvaruje robota do pozice, jaká mu vyhovuje, tak stačí přepnout pohony do stavu ON. Tento stav se projeví tak, že místo nápisu OFF, bude u příslušného pohonu napsáno číslo, reprezentující jeho pozici.



Obr. 2.5 – Příprava k uložení pozice robota

Abychom přenesli informaci o jednotlivých pozicích pohonných jednotek do paměťových míst, je potřeba stisknou tlačítko s zelenou šipkou v příslušném směru. V tomto případě bylo stisknuto tlačítko ze směru <Pose of Robots> na <Pose of Step>. Proto na následujcím obrázku 2.6 jsou uvedeny stejné hodnoty jak ve sloupci <Pose of Robots>, tak i ve sloupci <Pose of Step>.

Pozice STEP 0, která je definovaná řadou čísel ve sloupci <Pose of Robots> nebo ve sloupci <Pose of Step>, reprezentuje pozici, která je zobrazena na obrázku 2.3 a).

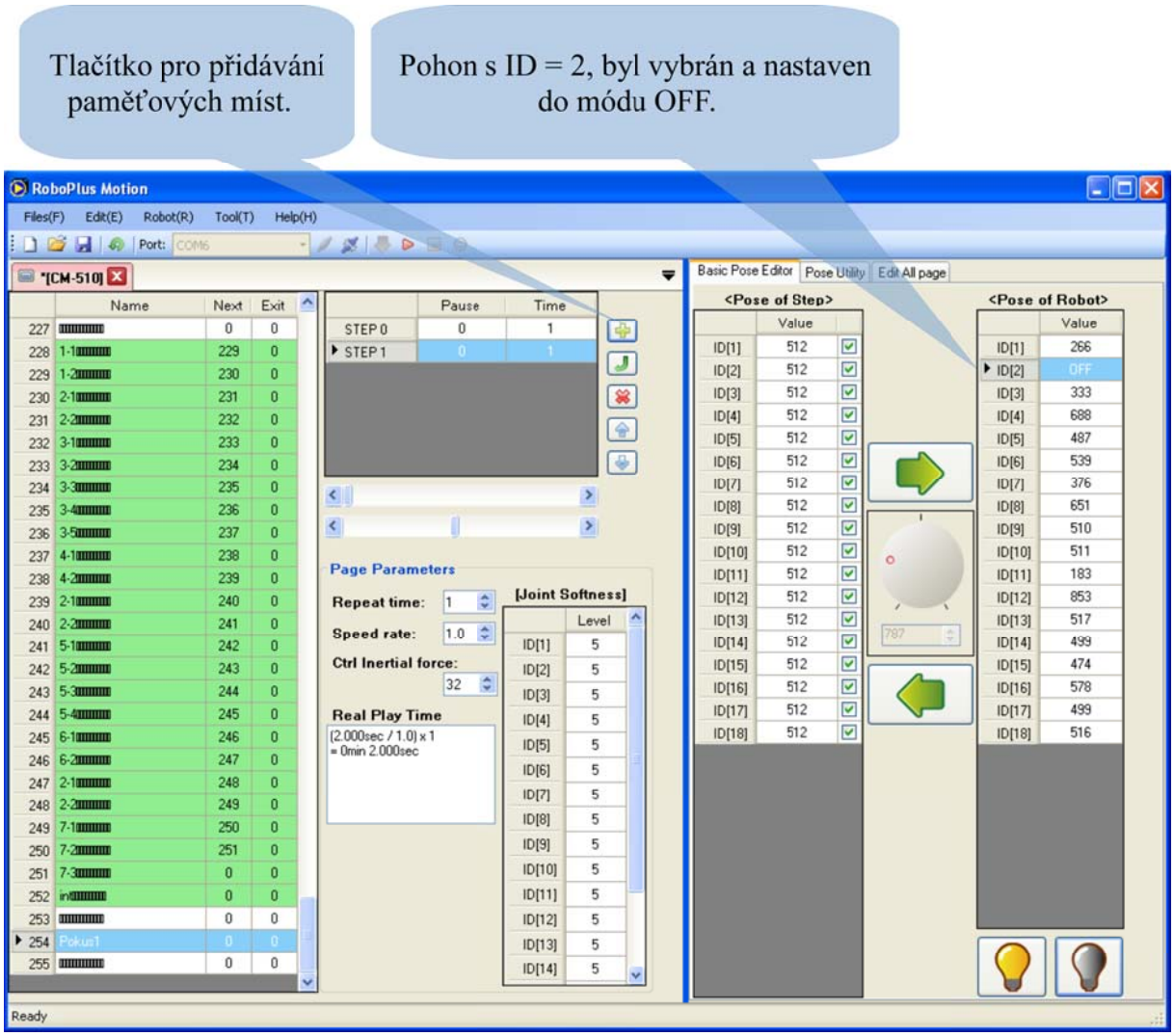


Obr. 2.6 – Uložená pozice robota s názvem STEP 0

Na obrázku 2.6 je vidět ukončená pozice, která je označena STEP 0 se zvoleným názvem skupiny Pokus 1. Na následujícím obrázku 2.7 je ukázka vytvoření následující pozice, která je označena STEP 1.

Vytvoření nových paměťových míst pro novou pozici se provádí klikem na ikonu se znaménkem plus, což způsobí vygenrování nových paměťových míst ve sloupci <Pose of Step>. V tomto sloupci jde vidět, že defaultní hodnota v těchto paměťových místech je 512, což znamená středovou pozici pohonných jednotek.

Dále je na tomto obrázku vidět, že byl vybrán pohon s ID = 2 a následně byl nastaven do módu OFF, což umožňuje ručně nastavit změnu pozice. V tomto případě se jedná o pohon v rameni a díky tomuto vypnutí je možné nastavit robota do pozice, která je ukázána na obrázku 2.3 b).

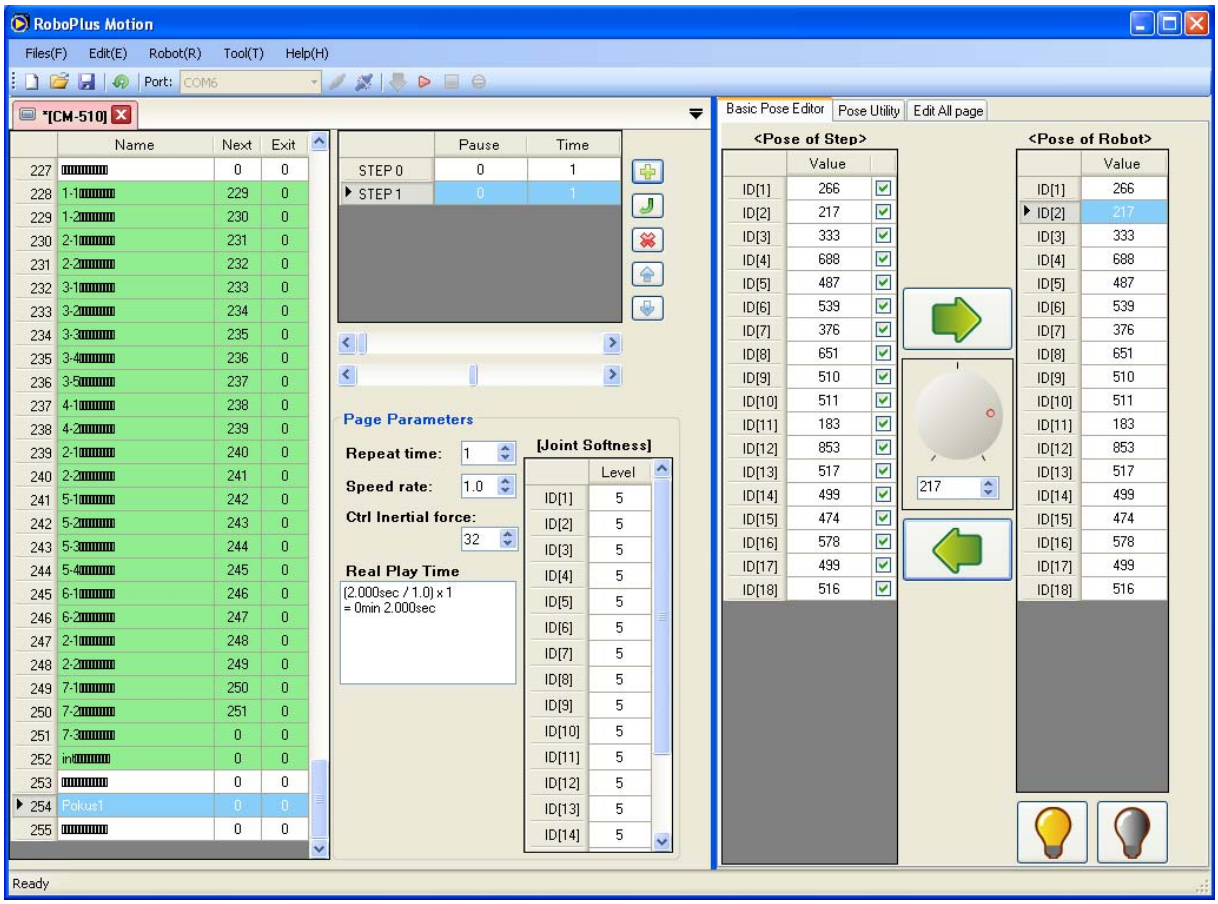


Obr. 2.7 – Tvorba nové pozice robota s názvem STEP 1

Následující obrázek ukazuje dokončenou pozici s názvem STEP 1. To, že je tato pozice dokončená, je možné zjistit, tak, že sloupce <Pose of Robots> a <Pose of Step> jsou shodné.

Aby tento stav byl dosažen, musel být pohon s ID = 2 nastaven zpět do módu ON a dále musely být, pomocí tlačítka, se zelenou šipkou, přesunuty informace ze sloupce <Pose of Robots> do sloupce <Pose of Step>.

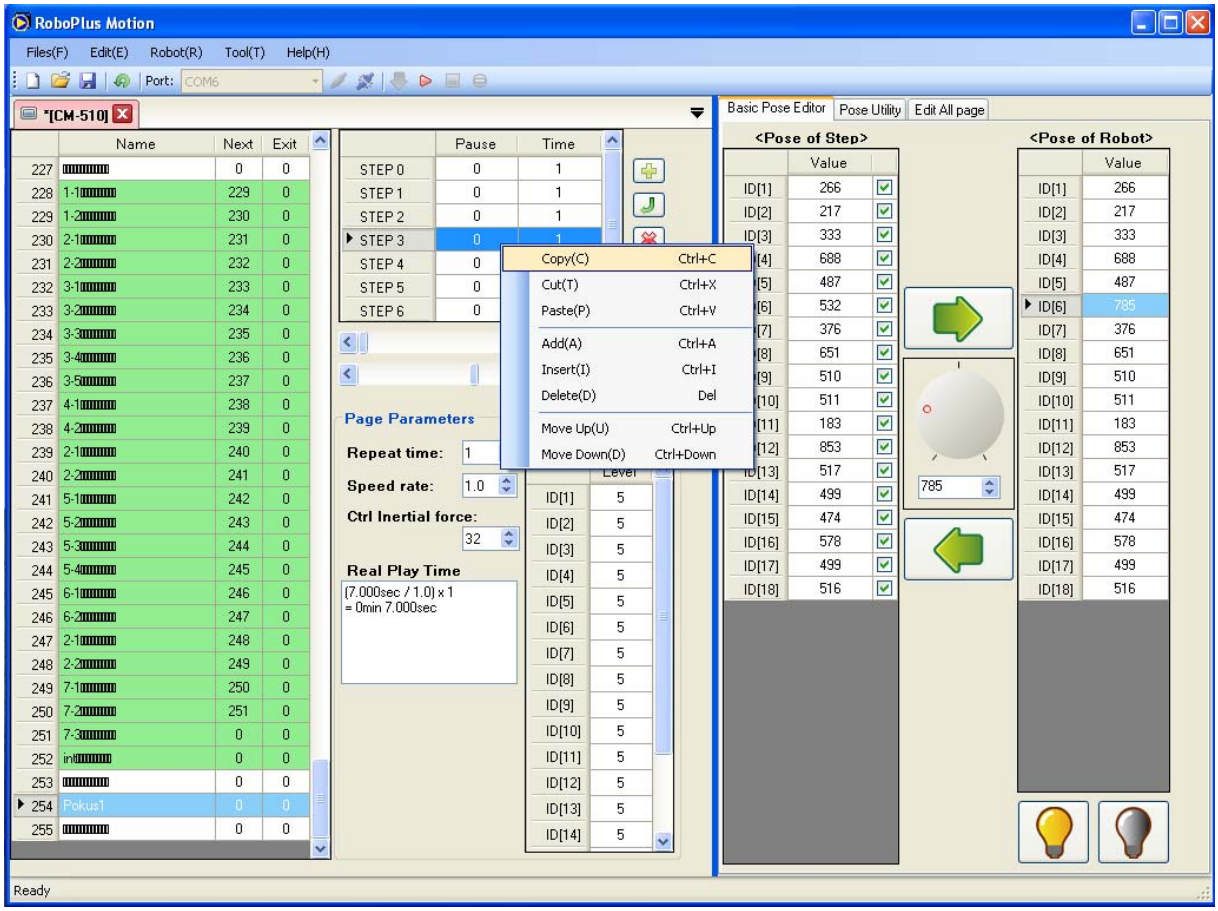
Tímto způsobem lze definovat libovolně složité pohyby, které vždy musí být rozděleny na jednotlivé pozice. Pro jednu skupinu pozic, v našem případě jsme si zvolili název Pokus 1, je možné vytvořit max. sedm pozic. Pokud chceme tvořit pohyby, jenž obsahují i více pozic, je to možné skládáním jednotlivých skupin pozic. Více na následující straně.



Obr. 2.8 – Dokončená pozice robota s názvem STEP 1

Co se týče složitějších pohybů, tj. pohybů, které jsou tvořeny více než jen sedmi pozicemi, je možné využít další vlastnost tohoto software. Pokud potřebujeme realizovat takový komplikovaný pohyb, tak je možné jej vytvořit, tak, že se vytvoří několik skupin pohybů, které navzájem budou na sebe navazovat. To, aby se pak pohyb vykonal jako jeden celek, je možné nadefinovat v levé části okna, v položce Next.

Zde, na tomto obrázku 2.9, je vidět dokončený pohyb mávání rukou Humanoidního robota. Na realizaci tohoto pohybu je použito sedm pozic. Během práce se často setkáváme s potřebou různým způsobem měnit pořadí jednotlivých pozic, případně přidávat či ubírat jednotlivé pozice. Z tohoto důvodu je obrázek 2.9 zobrazen s pop-up oknem, které ukazuje, že všechny zmíněné operace s paměťovými místy jsou možné.



Obr. 2.9 – Dokončený pohyb robota s názvem Pokus 1

3. Doplňkové hardwarové komponenty

ČAS KE STUDIU: 100 minut

CÍL

Studiem následující kapitoly se čtenář dozví, jaké snímače můžou být použity s roboty, jejichž návody na sestavení jsou uvedeny v následující kapitole.

VÝKLAD

Systém Bioloid je otevřený systém, který umožňuje připojení libovolného snímače, v případě, že je dodržen komunikační protokol, který je užíván na sběrnici. To znamená, že je možné si vybrat snímač, doplnit jej o řídicí jednotku s příslušným řídicím programem a takto upravený snímač připojit na sběrnici řídicí jednotky CM-xx. Příkladem takového snímače, jenž není výrobkem firmy Robotis, je modul HaViMo, který je popsán v kapitole 3.4.

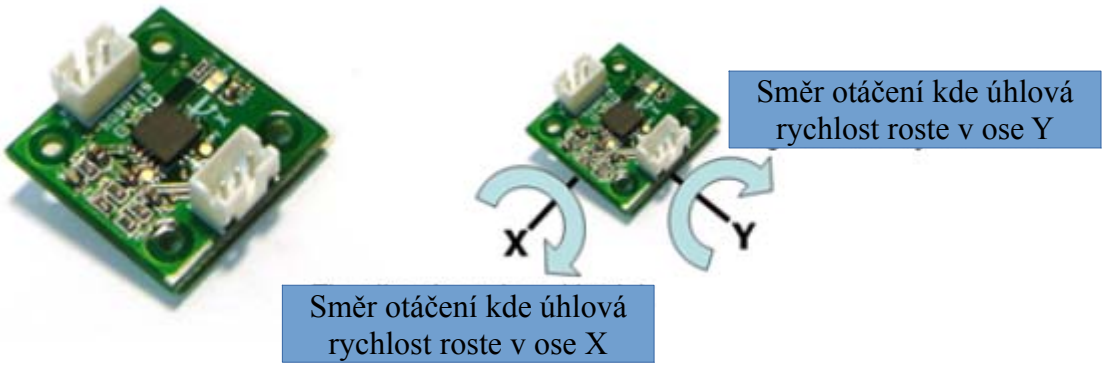
Pokud chceme využít již hotových snímačů, které dodává přímo firma Robotis k tomuto systému, tak máme možnost využít různých tvarů snímače vzdálenosti od překážky (tzv. IR snímač, viz. kapitola 3.2), dále pak gyroskopický (viz. kapitola 3.1) snímač, dotykové snímače a také zvukové a světelné detektory.

Kromě základních snímačů je v této kapitole také uveden popis bezdrátového dálkového ovládače pro snadnější manipulaci s roboty, viz. kapitola 3.3.

3.1. Gyroskopický snímač

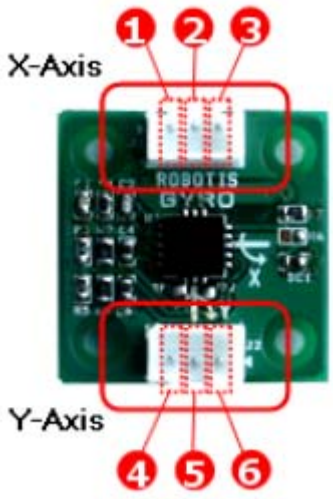
Gyroskopický snímač je používán u robota typu Humanoid. Snímač u tohoto robota hraje významnou roli při chůzi, kdy tento snímač funguje pro realizaci korekce chůze. Díky různým materiálům a nerovnostem na povrchu, kde se robot pohybuje, vznikají situace, kdy robot mění své těžiště, což bez korekce pohybu může vést k pádu.

Zmíněný gyroskopický snímač je polovodičová součástka, která reaguje úhlovou rychlost ve dvou osách. Pozor, díky této vlastnosti je možné sledovat změny pohybu robota, nikoliv jeho pozice.



Obr. 3.1 – Gyroskopický snímač dodávaný firmou Robotis

Jak lze vidět na obrázku 3.1, tento snímač má dva tří-pinové konektory. I když tyto konektory mají tři piny, tak z hlediska funkčnosti, jejich úloha je zcela odlišná od tří-pinových konektorů, které jsou na pohonných jednotkách. Proto tyto snímače lze připojit pouze k řídicím jednotkám typu CM-510, CM-530 a CM-700. To z důvodu, že tyto řídicí jednotky mají speciální konektor pro připojení analogových snímačů.



Obr. 3.2 – Označení pinů gyroskopického snímače

Piny Osy X (X - Axis)
1: ADC, Analogový signál reprezentující úhlovou rychlost
2: GND
3: Vcc (+5V)

Piny Osy Y (Y - Axis)
4: Vcc (+5V)
5: GND
6: ADC, Analogový signál reprezentující úhlovou rychlost

Následující ukázka kódu je určena pro RoboPlus Task. V této ukázce se předpokládá, že gyroskop je připojen na port 5 a 6. Program reprezentuje možnost načtení deseti hodnot z každé osy gyroskopu a to hodnoty zprůměruje. To může sloužit jako základní filtrace signálů z gyroskopu.

ŘEŠENÉ PŘÍKLADY

```
FBBalData = PORT[6]
RLBalData = PORT[5]


FBBalCenter = 0
RLBalCenter = 0

LOOP FOR ( i = 1 ~ 10 )
{
    FBBalCenter = PORT[6]
    RLBalCenter = PORT[5]

    FBBalCenter = FBBalCenter + FBBalData
    RLBalCenter = RLBalCenter + RLBalData

    Timer = 0.128sec
    WAIT WHILE ( Timer > 0.000sec )
}

FBBalCenter = FBBalCenter / 10
RLBalCenter = RLBalCenter / 10
```

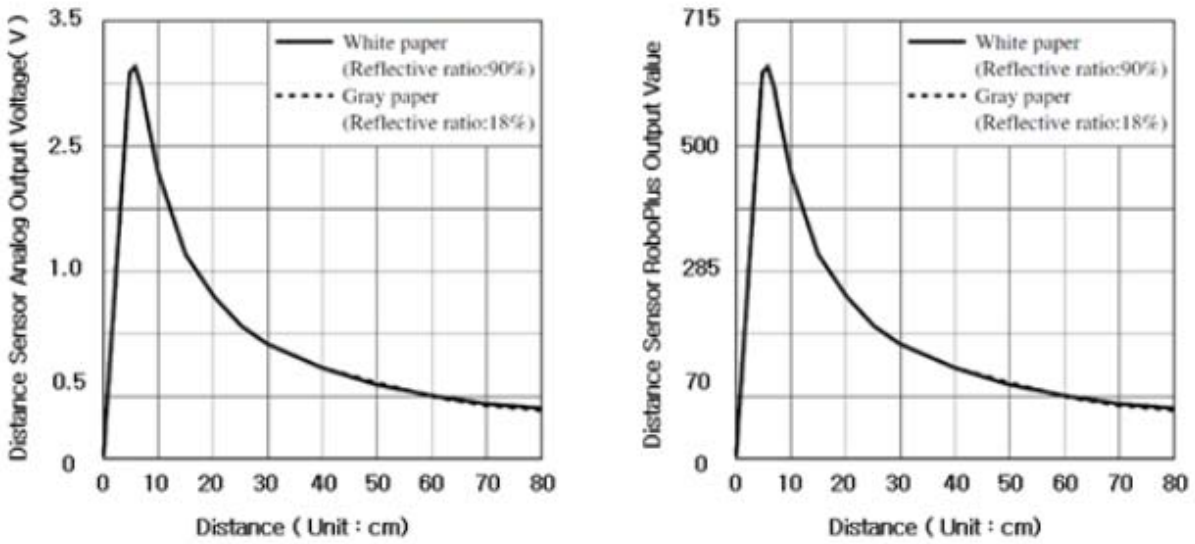
Obr. 3.3 – Ukázka kódu pro čtení hodnot z gyroskopu

3.2. IR senzor vzdálenost

Infračervený snímač vzdálenosti funguje na principu vysílání infračerveného signálu, který při odrazu dopadne zpět na snímač a z této doby se vypočítává vzdálenost překážky od snímače. To znamená, že takový senzor má jednu vysílací IR diodu a přijímací IR tranzistor.

Obr. 3.4 – Ukázka IR senzoru vzdálenosti

Protože tento snímač používá elektromagnetické záření s podobnou vlnovou délkou jakou má viditelné světlo, je možné říct, že i tento signál má podobné vlastnosti jako světlo. Základní vlastností je částečná citlivost na barvu předmětu / překážky. Vzhledem k tomu, že bílá barva má vlastnost odrážet většinu vlnových délek elektromagnetického záření a černá barva má naopak schopnost pohlcovat, tak i zde je nutné očekávat změnu charakteristiky chování tohoto snímače. Proto následující charakteristika snímače je uvedena pro bílý papír s odrazivostí 90% a pro porovnání také pro šedý papír, který odráží jen 18%.



Obr. 3.5 – Statická charakteristika IR senzoru vzdálenosti

3.3. Bezdrátový ovládač

Bezdrátový ovládač s označením RC-100 / RC-100A je určený k ovládání robotů. Tento dálkový ovládač je schopný generovat pouze binární informace pomocí deseti funkčních tlačítek.

Obr. 3.6 – Dálkový ovládač RC-100 / RC-100A

Některá z těchto tlačítek však mohou mít rozdílnou funkci, v závislosti na módu v jakém je daný ovládač používán. Ovládač je standardně vybaven IR vysílačem. To znamená, že přenos informace je velice podobný způsobu ovládání televize.

Obr. 3.7 – Princip přenosu dálkovým ovládačem RC-100 / RC-100A

Aby tento přenos mohl proběhnout, je potřeba aby řídicí jednotka měla připojený přijímač, který je na obrázku 3.7 označen, jako IR-Receiver. Po stisknutí prostředního tlačítka, které slouží pouze k zapnutí / vypnutí vysílače, stačí jen stisknout tlačítko a informace o stisknutém tlačítku se přenese přes IR-Receiver do řídicí jednotky.

V případě, že v jednom prostoru bude chtít ovládat roboty více lidí, mohlo by docházet ke vzájemnému ovlivňování řídicích povelů. Proto každý vysílač je vybaven možností změnit tzv. vysílací kanál. Změna vysílacího kanálu je ukázána na následující stránce.

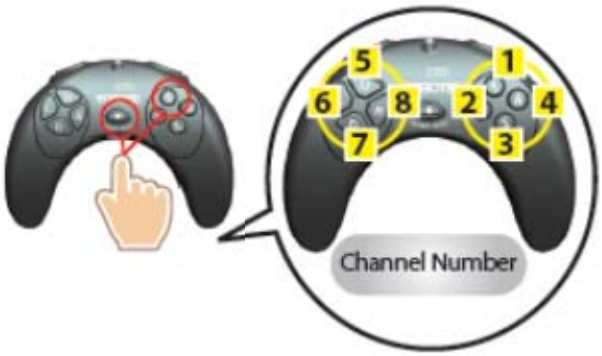
Změna vysílacího kanálu vysílače RC 100 / RC 100A

Vysílací kanál je možné nastavit na hodnotu **1** – **8**. tj. pomocí tohoto ovládače, pomocí IR přenosu informace, je možné řídit současně osm různých robotů. Ke zjištění aktuálně nastaveného vysílacího kanálu slouží tlačítka, která jsou označena číslem 5 a 6. Z tohoto důvodu tato tlačítka nelze použít k řízení.

Pro nastavení vysílacího kanálu stačí provést tři kroky. V prvním kroku je nutné zapnout vysílač stisknutím tlačítka POWER – obrázek 3.8.

Obr. 3.8 – Zapnutí ovládače RC-100 / RC-100A

V druhém kroku je nutné zmáčknout tlačítko POWER a zároveň jedno z tlačítek reprezentující číslo kanálu. Pozor, tuto kombinaci držet přibližně jednu sekundu.



Obr. 3.9 – Volba vysílacího kanálu RC-100 / RC-100A

V případě, že kanál je nastaven, dojde z zablikání LED podle zvoleného čísla kanálu.

Obr. 3.10 – Počet bliknutí reprezentuje zvolený vysílací kanál

V případě, že ovládač má nastaveno číslo vysílacího kanálu a my ho potřebujeme zjistit, je možné stisknout tlačítka označená 5 a 6. Tím dojde opět k zablikání LED a počet zablikání reprezentuje číslo vysílacího kanálu.

Obr. 3.11 – Zjištění aktuálně nastaveného čísla kanálu

Aby řídicí jednotka mohla změnu vysílacího kanálu akceptovat, je nutné někde na začátku programu nastavit vnitřní proměnnou **RC-100 Channel** na požadované číslo kanálu. Například **RC-100 Channel = 3**, znamená, že řídicí jednotka bude přijímat informace pouze z vysílače, který vysílá na kanále 3.

ŘEŠENÉ PŘÍKLADY

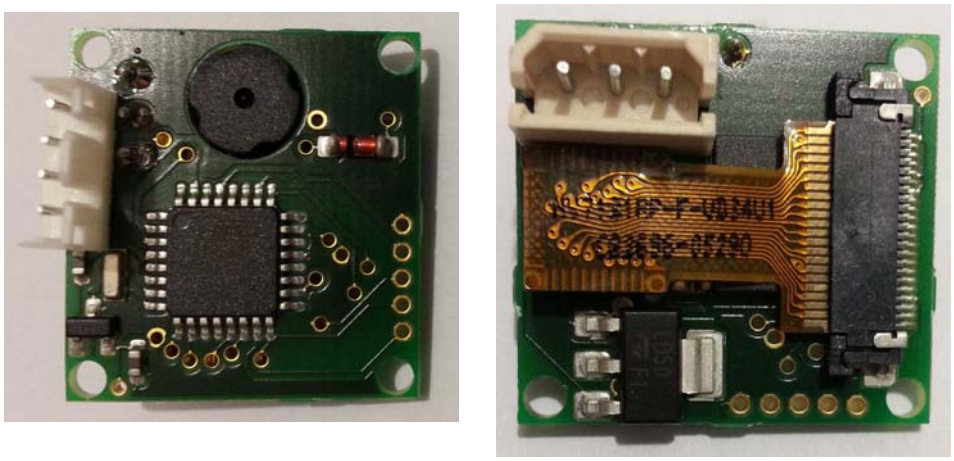
```
1   START PROGRAM
2   {
3       // It set up the RC-100 channel.
4       IF ( Button count <= 8 )
5       {
6           RC-100 Channel = Button count
7       }
8       ELSE
9       {
10          RC-100 Channel = 8
11      }
12  }
```

Obr. 3.12 – Ukázka programu, který podle počtu stisku tlačítka START nastaví číslo kanálu

3.4. Modul HaViMo 2.0

HaViMo je externí modul, který obsahuje CMOS kameru a procesor implementující některé algoritmy z oblasti zpracování obrazu. Jedná se o algoritmy, které jsou známy pod názvy: BLOB Tracking, Region Growing, a Image Girding.

Tento modul je vytvořen tak, aby jej bylo možné připojit k systému Bioloid, Robobuilder nebo jen k embedded systému s malým výpočetním výkonem. Ke komunikaci s nadřazenými řídicími jednotkami slouží UART (sériová linka – half duplex) s úrovni TTL.



Obr. 3.13 – Pohled z obou stran na modul HaViMo 2.0

Základní funkcí modulu je rozpoznávání a sledovaní barevných oblastí. To lze využít například pro realizaci robota, který dokáže identifikovat pozici objektů podle zvolené barvy. Například robotický fotbal je často založen na hledání míčku, který má odlišnou barvu, než je barva pozadí (hřiště).

Obr. 3.14 – Snímek červeného a zeleného míčku, který je pořízen CMOS kamerou

Je důležité si uvědomit, že modul HaViMo 2.0 nedokáže rozpoznat tvary, tak jako lidský mozek. Tento modul dokáže najít pouze plochy zvolených barev v zorném poli kamery a na požádání z nadřazeného systému, je schopen vrátit obdélníkové souřadnice této plochy. Tento modul dokáže najednou identifikovat až sedm barevných skupin. (Při realizaci robotického fotbalu však stačí rozlišit pouze barvu míčku...) Zde se hovoří o barevných skupinách z toho důvodu, že například červená kulička na obrázku 3.14 je tvořena několika odstíny červené barvy, které postupně přecházejí až v bílou barvu.

Pokud je žádoucí využít tohoto modulu k realizaci robota, je nutné, tento modul nejprve nakonfigurovat na požadované parametry.

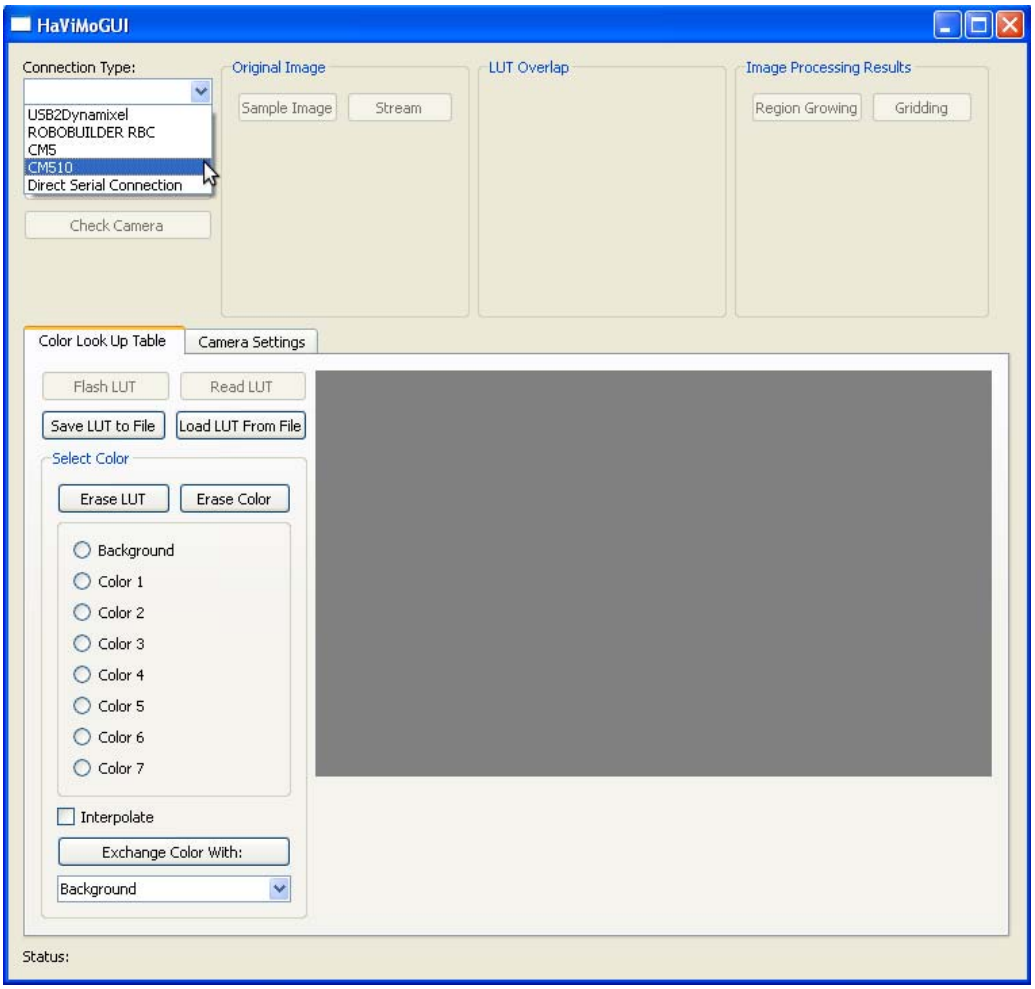
Konfigurace modulu HaViMo 2.0

K tomu aby mohla být konfigurace úspěšně provedena je nutné zajistit kvalitní a stabilní světelné podmínky snímané scény. Dále je nutné mít nainstalovaný software HaViMoGUI, který je možné zdarma stáhnout a používat na operačních systémech Windows XP a výše. Software je k dispozici na adrese: http://sourceforge.net/projects/havimo/

Konfigurace se skládá z několika následujících kroků:

1. *Připojení modulu k PC*

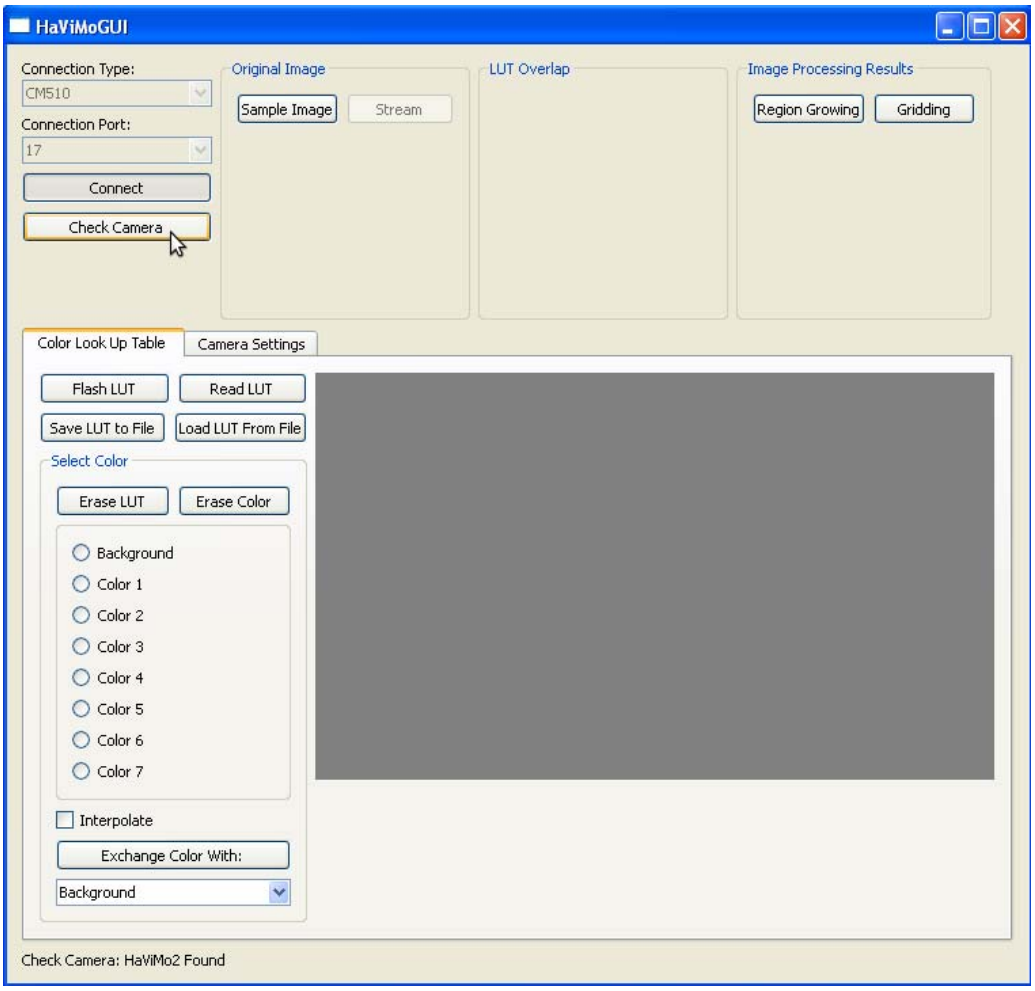
Připojit HaViMo modul je možné několika způsoby, zde však budeme předpokládat, že budeme používat připojení skrze řídicí jednotku CM-5 / CM-510. Řídicí jednotku CM-530 je možné použít také ke konfiguraci, avšak k tomuto účelu je nutné dočasně změnit firmware CM-530. Viz. následující odstavec – *Změna firmware CM-530*.



Obr. 3.15 – Nastavení typu připojení HaViMo modulu k PC

2. *Otevření komunikačního portu a ověření dostupnosti modulu (Check Camera)*

Po výběru příslušného komunikačního portu (COM x) a stisku tlačítka „Connect“, dojde k otevření komunikačního portu. Dále je nutné ověřit zda-li software HaViMoGUI je schopen komunikovat s modulem. To lze ověřit tlačítkem „Check Camera“. Pokud je vše v pořádku, bude ve stavovém řádku napsáno: „HaViMo2 Found“. Jak lze vidět na obrázku 3.16.



Obr. 3.16 – Úspěšné připojení HaViMo modulu k PC

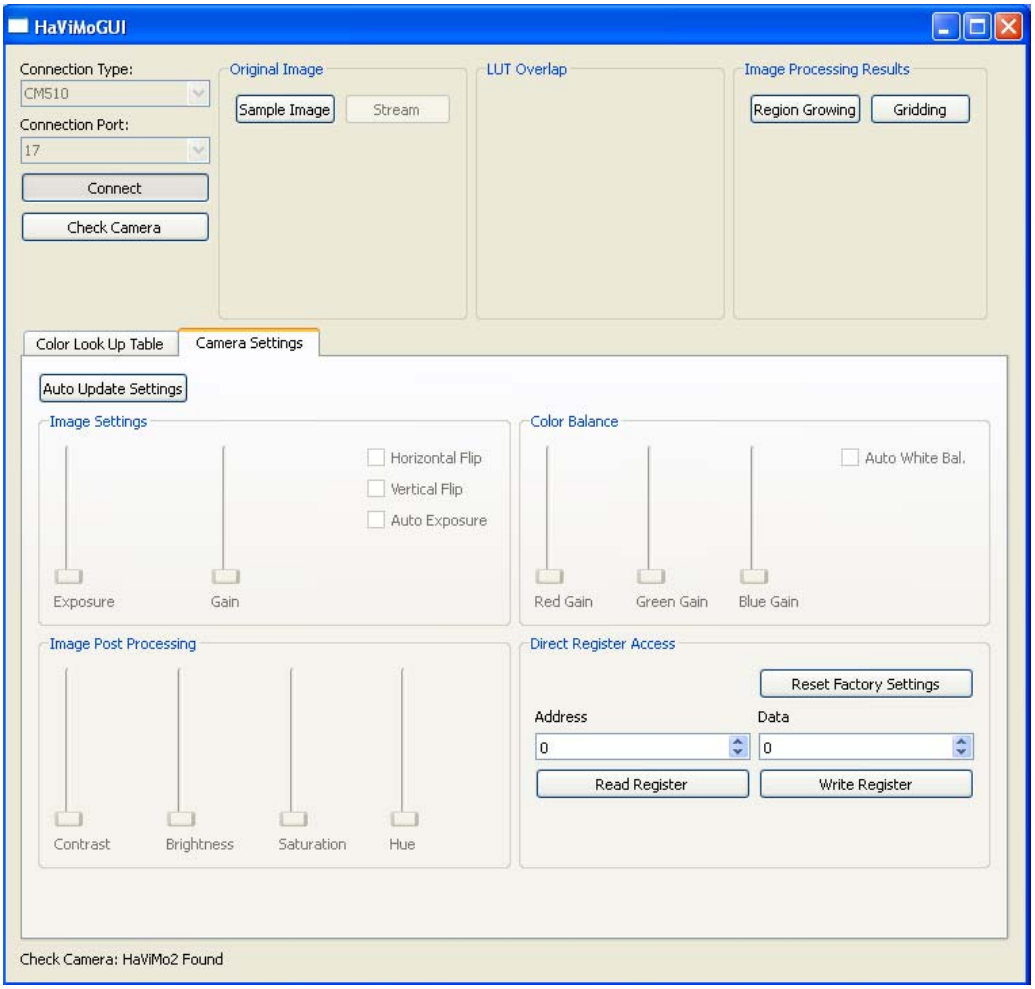
V opačném případě bude vypadat nápis ve stavovém pruhu, takto: „Not Found“.

Upozornění!!!
Pro úspěšné připojení je potřeba nejprve aktivovat řídicí jednotku CM-510 a spustit mód Manage. (Bude svítit zelená LED) Až poté je možné stisknout tlačítko „Connect“ v software HaViMoGUI. Pokud tento postup nebude dodržen, je možné, že komunikační port bude otevřen, ale modul HaViMo nebude nalezen.
U řídicí jednotky CM-5 mód Manage spuštěn být může, ale nemusí. Stačí když bude LED blikat na pozici Manage Mode.

3. Nastavení optických parametrů kamery (Camera Settings)

Vzhledem k různým možnostem osvětlení snímané scény, např. barva světla – bílá barva může vypadat mírně do žluta, různá intenzita osvětlení, apod. Proto je potřeba nejprve nastavit kameru na tyto různé podmínky. Aby bylo možné parametry měnit je potřeba stisknout tlačítko „Auto Update Settings“. Tímto se aktivuje karta s modifikovatelnými parametry.

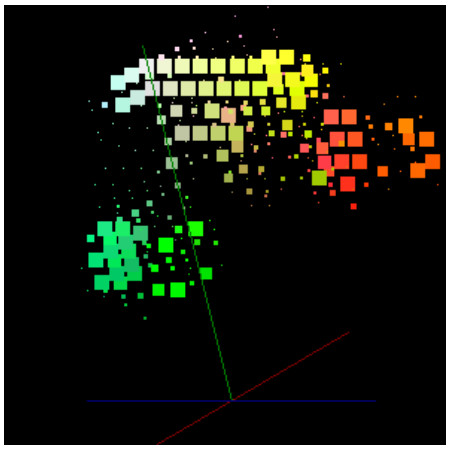
Na kartě „Camera Settings“, viz. obrázek 3.17 je možné nastavit:
- Automatické vyvážení bíle (Auto White Balance)
- Automatický čas expozice (Auto Exposure)
- Převrácení obrazu podle Horizontální/Vertikální osy (Horisontal/Verical Flip)
- a další...



Obr. 3.17 – Nastavení optických parametrů na kartě „Camera Settings“

4. Načtení zkušebního snímku (Sample Image)

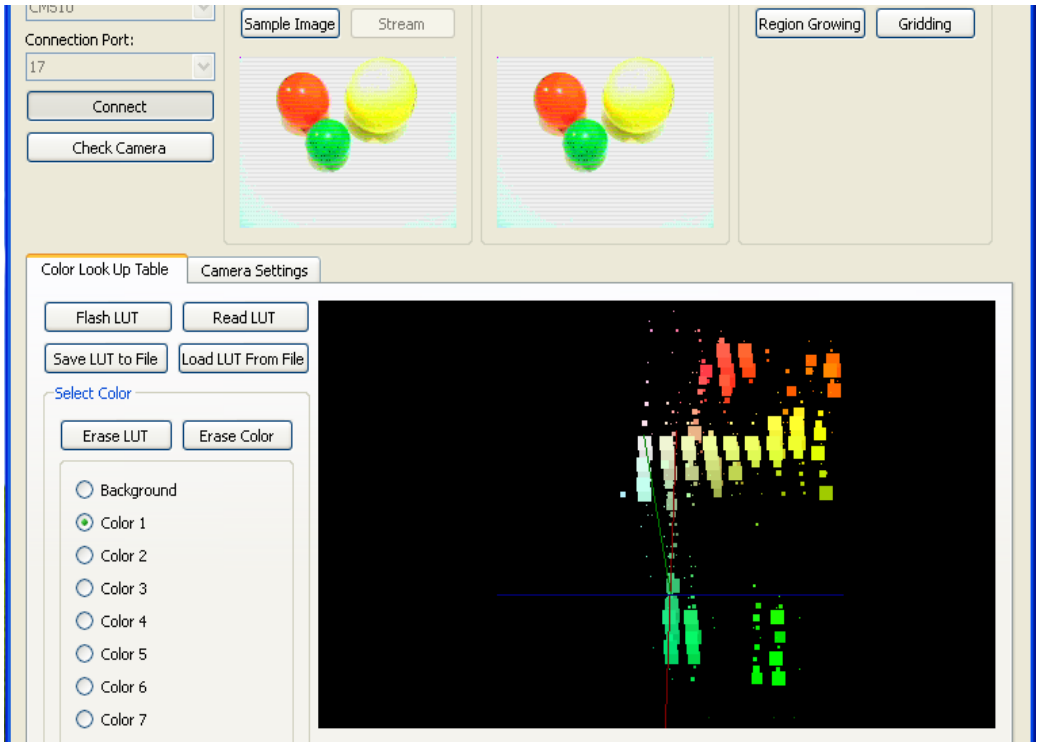
Pro ověření správnosti nastavení je možné načíst snímek scény (stiskem tlačítka „Sample Image“), která je v zorném poli kamery. Načtení jednoho snímku trvá několik sekund. Po tomto procesu software HaViMoGUI vytvoří 3D RGB model načteného snímku, viz. obrázek 3.18.



Tento obrázek je tvořen třemi osami, kde každá osa reprezentuje jednu barvu barevného modelu RGB. Pro lepší orientaci v 3D RGB modelu, je možné pomocí myši měnit úhel pohledu na tento 3D model. Na obrázku 3.18 jsou také zobrazeny barevné čtverce, jejichž velikost reprezentuje četnost výskytu pixelů s daným odstínem.

Obr. 3.18 – 3D RGB model načteného snímku

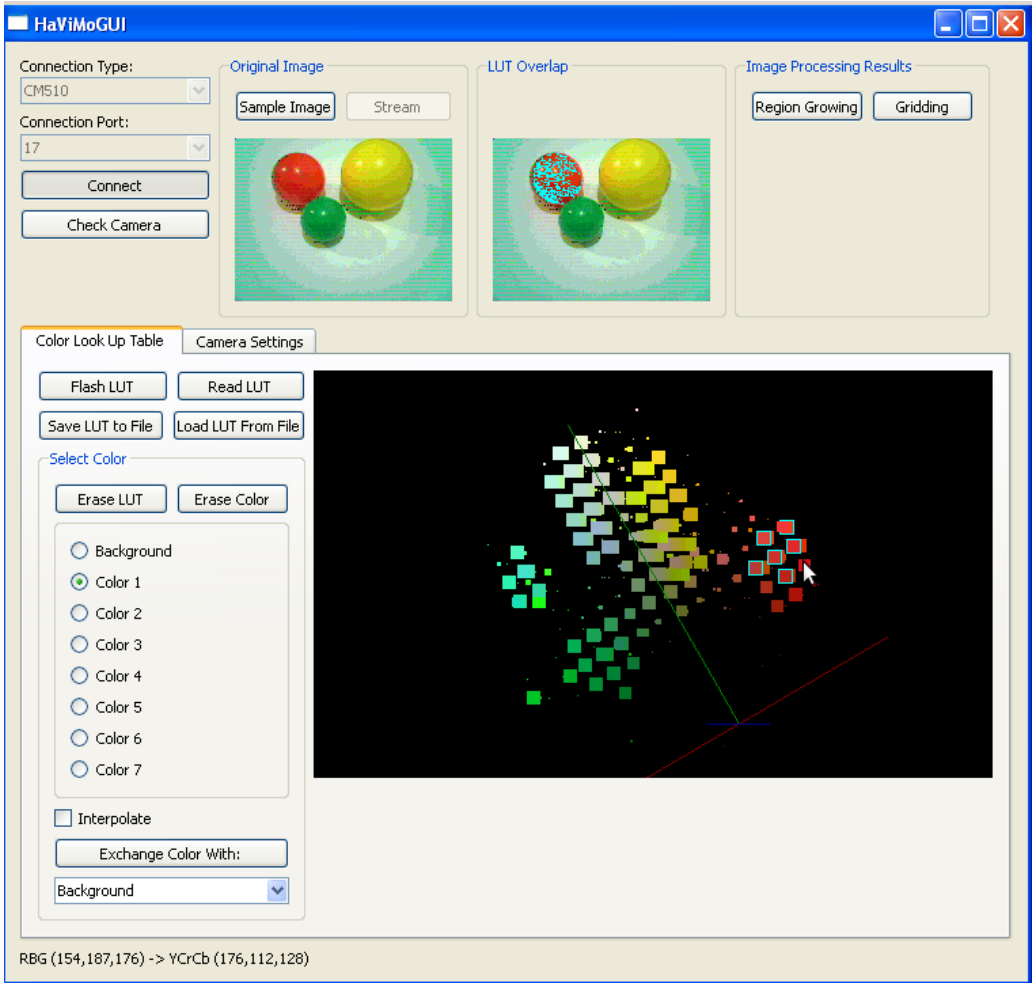
Následující obrázek 3.19 ukazuje jiný pohled na 3D RGB model, jenž je zobrazen na obrázku 3.18. Dále také ukazuje snímek, ze kterého byl daný model vytvořen.



Obr. 3.19 – 3D RGB model načteného snímku tří barevných kuliček

5. *Definice barevných skupin*

Jak již bylo napsáno, modul HaViMo 2.0 dokáže v rámci jednoho nastavení rozlišovat sedm barevných skupin. Nastavení jednotlivých barevných skupin se provádí výběrem dané skupiny (stačí kliknout levým tlačítkem myši na kruhové tlačítka), které jsou označeny nápisem „Color 1“ až „Color 7“.



Obr. 3.20 – Nastavení barevné skupiny s označením „Color 1“

Obrázek 3.20 ukazuje označení kruhového tlačítka u barevné skupiny s označením „Color 1“. Této barevné skupině je důležité přiřadit veškeré odstíny, které chceme do této skupiny zařadit. To provádíme postupným klikáním na jednotlivé barevné čtverce z 3D RGB modelu nebo postupně klikáme na vybraný objekt v levém obrázku, který je umístěn ve skupině „Original Image“.

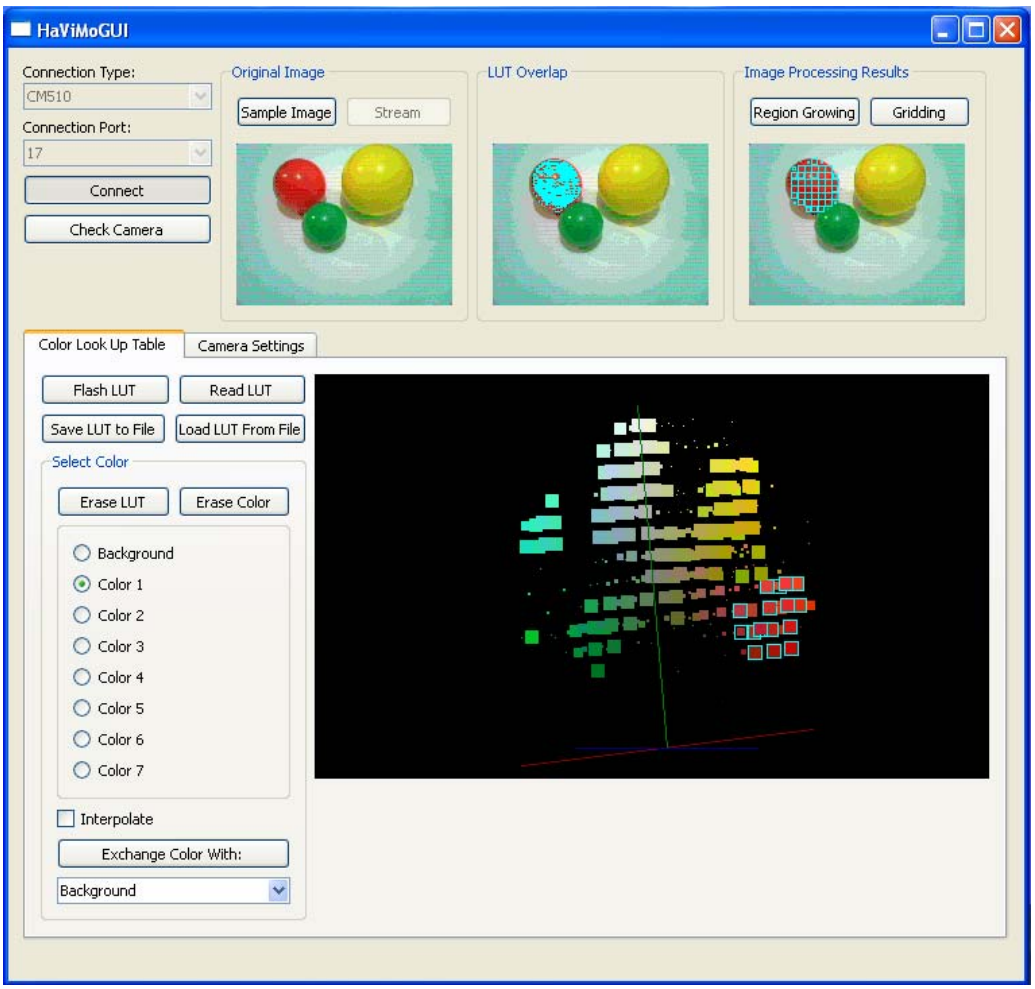
Výběr jednotlivých odstínů vybíráme pokud možno tak, aby se jednalo o pixely, které se nacházejí na objektu, který je středem našeho zájmu. Zda-li tomu tak opravdu je, můžeme sledovat na obrázku umístěném vpravo, tj. ve skupině s označením „LUT Overlap“.

6. *Uložení konfiguračních parametrů (Flash LUT)*

Aby modul HaViMo mohl realizovat své výše zmíněné algoritmy, je nutné vybrané barevné skupiny uložit do paměti procesoru, který je součástí modulu. To se provede stiskem tlačítka „Flash LUT“. Tato operace vyžaduje k provedení několik desítek sekund, avšak po úspěšném provedení jsou tyto informace zapsány v trvalé paměti procesoru (Flash EPROM), což znamená, že konfiguraci nebude nutné provádět po každém odpojení/připojení napájení.

7. *Ověření nastavení (Region Growing / Gridding)*

Správné nastavení a uložení konfigurace je možné ověřit pomocí tlačítek „Region Growing“ nebo „Gridding“. Po stisku tlačítka „Gridding“ se vykreslí oblast, která odpovídá zvolenému nastavení. Tato oblast se vykreslí ve skupině s názvem „Image Processing Results“, jak lze vidět na obrázku 3.21.



Obr. 3.21 – Ověření správnosti nastavení modulu

8. *Uzavření komunikačního portu / odpojení*

Pokud je vše nastaveno správně, stačí odpojit komunikační port. To lze provést opětovným kliknutím na tlačítko „Connect“. Tím se tlačítko takzvaně odmáčkne. Pokud by tlačítko „Connect“ zůstalo zamáčknuté, nedošlo by k uvolnění komunikačního portu a tím by pravděpodobně nebylo možné pokračovat při další práci se softwary RoboPlus Studia.

Úspěšnou realizací těchto kroků, lze pokládat modul HaViMo 2.0 za nastavené a je možné jej propojit s konstrukcí nějakého robota. Aby daný robot mohl využít možnosti HaViMo modulu, musí být vytvořený příslušný *.tsk soubor, který bude definovat jaké informace budou využívány z HaViMo modulu a také k čemu. Jedna ze základních úloh je popsána v odstavci – **Příklad použití**.

Změna firmware CM-530, pro konfiguraci HaViMo modulu

Modul HaViMo byl vytvořen v době kdy existovala řídicí jednotka CM-5 a CM-510. Momentálně je na trhu nová řídicí jednotka CM-530, která není uvedena na seznamu možných způsobů připojení v software HaViMoGUI. Proto byl dodatečně vytvořen speciální firmware pro CM-530, který umožňuje tuto jednotku využít ke konfiguraci HaViMo modulu, tak jako by se pracovalo s řídicí jednotkou CM-510.

Firmware je tvořen jediným souborem s příponou *.hex a je dostupný na adrese:
http://www.havisys.de/wp-content/uploads/2012/12/CM530.zip

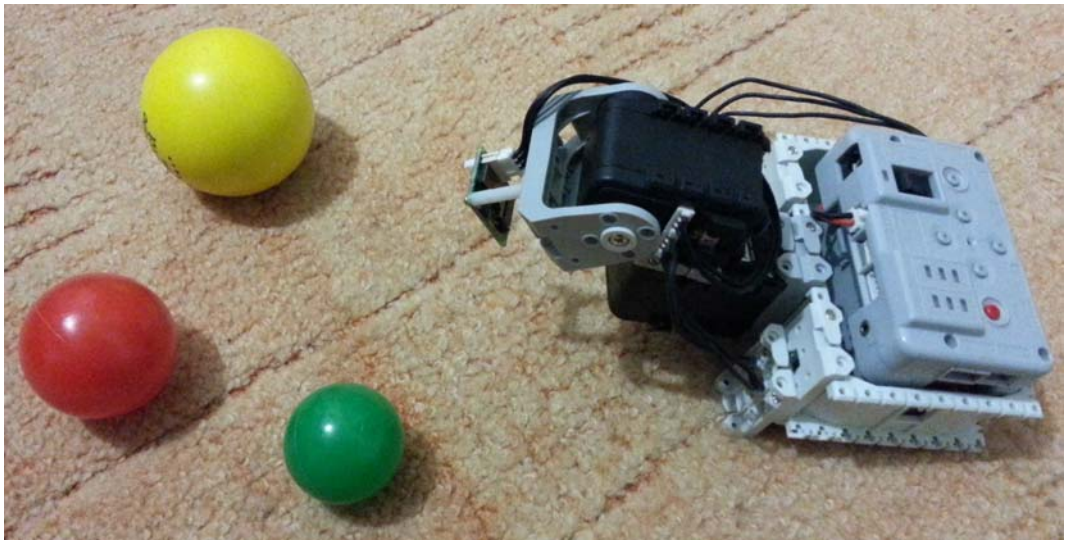
Po stažení daného souboru a rozbalení archívu stačí daný hex soubor zavést do řídicí jednotky standardním způsobem. K tomuto účelu je možné využít software RoboPlus Terminál a postupuje se podobně, jako bychom si vytvořili vlastní program v jazyce C.
Po spuštění konfiguračního nástroje HaViMoGUI, je nutné vybrat položku CM-510 v kategorii typ připojení a to i přesto, že používáme CM-530.

Důležité je poznamenat, že tento firmware je vytvořený pouze pro konfiguraci HaViMo modulu se softwarem HaViMoGUI. Proto po provedení konfigurace je potřeba obnovit původní firmware v řídicí jednotce pomocí programu RoboPlus Manager. Až poté bude možné opět nahrávat tsk a případně mtn soubory do řídicí jednotky.

ŘEŠENÉ PŘÍKLADY - Modul HaViMo použitý pro sledování pohybu míčku

Tato úloha byla vytvořena pro ukázku základního využití HaViMo modulu ve spojení s pohony AX-12A. Úloha obsahuje dva pohony AX-12A, které jsou mechanicky propojené tak aby modul HaViMo bylo možné natáčet ve dvou navzájem kolmých osách.

Nejprve byla provedena konfigurace modulu HaViMo na tři skupiny barev. Na červené, zelené a žluté odstíny, které reprezentují tři barevné koule, jak lze vidět na obrázku 3.22.



Obr. 3.22 – Natáčecí mechanizmus modulu HaViMo

Červené odstíny byly uložené ve skupině s názvem „Color 1“, zelené a žluté odstíny pak ve skupinách s názvy „Color 2“ a „Color 3“. Níže je uvedený výpis programu z tsk souboru, který má za úkol nastavit takovou pozici HaViMo modulu, aby pokud možno byl hledaný objekt vždy ve středu zorného pole kamery. Program je vytvořen tak aby reagoval pouze na jednu barvu. To znamená, že při pohybu kuličkou, v našem případě červenou, bude docházet k otáčení HaViMo modulu ve směru pohybu červené koule.

Program však nezohledňuje rychlost pohybu koule, proto při rychlejším pohybu se může stát, že natáčecí systém nestačí zareagovat a červená koule zmizí ze zorného pole kamery. Čímž se vykonávání algoritmu zastaví, protože pro danou barvu nebude HaViMo modul posílat souřadnice s výskytem dané barvy.

Následující program je tvořen hlavní funkcí – START PROGRAM, která obsahuje smyčku, která se bude opakovat dokud nebude stisknuto tlačítko „D“ na řídicí jednotce. Po stisku tohoto tlačítka dojde k ukončení programu.

Hlavní součástí programu je funkce Get_Bounding_Box, která získává čtyři hodnoty (Min**x**, Max**x**, Min**y** a Max**y**), ze kterých se následně vypočte šířka a výška obdélníkové oblasti, která obsahuje barevný odstín definovaný v barevné skupině Color 1. Následně se vypočte střed této oblasti a pak už jen následuje řada podmínek, které určují, kterým směrem a o kolik se mají pohony pohnout aby se barevná oblast nacházela přibližně ve středu zorného pole kamery.

Výpis hlavního programu:

(Část tohoto programu je součástí: http://robosavvy.com/Builders/hamid_m/camera2.zip) Tento program byl přizpůsoben pro použití dvou pohonů AX-12A, kde pohon s ID 2, realizuje horizontální pohyb (v programu je použito označení Cx) a pohon s ID 1, realizuje vertikální pohyb (v programu je použito označení Cy).

```
START PROGRAM
{
    LOOP WHILE ( Button != D )
    {
        CamID  = 100
        Color  = 1
        ID[CamID]: ADDR[0(b)]  = 0
        CALL Get_Bounding_Box
        Cx  = Minx  +  Maxx
        Cy  = Miny  +  Maxy
        Cx  = Cx  /  2
        Cy  = Cy  /  2
        IF ( Max  !=  0 )
        {
            IF ( Cx  >  90 )
            {
                Cx  = Cx  -  80
                Cx  = Cx  -  5
                ID[2]: Goal position  = ID[2]: Present position  -  Cx
            }
            ELSE IF ( Cx  <  70 )
            {
                Cx  = 80  -  Cx
                Cx  = Cx  -  5
                ID[2]: Goal position  = ID[2]: Present position  +  Cx
            }
            ELSE IF ( Cy  <  60 )
            {
                Cy  = 70  -  Cy
                ID[1]: Goal position  = ID[1]: Present position  +  Cy
            }
            ELSE IF ( Cy  >  80 )
            {
                Cy  = Cy  -  70
                ID[1]: Goal position  = ID[1]: Present position  -  Cy
            }
        }
    }
    END PROGRAM
}
```

Výpis funkce Get_Bounding_Box:
(Tato funkce byla převzata z http://robosavvy.com/Builders/hamid_m/camera2.zip)

```
42 FUNCTION  Get_Bounding_Box
43 {
44     Max  = 0
45     LOOP FOR  ( Index  = 1  ~ 15  )
46     {
47         Addr  = Index  *  16
48         IF  ( ID[CamID]: ADDR[Addr(b)]  !=  0  )
49         {
50             Addr  = Addr  +  1
51             IF  ( ID[CamID]: ADDR[Addr(b)]  ==  Color  )
52             {
53                 Addr  = Addr  +  1
54                 Size  = ID[CamID]: ADDR[Addr(w)]
55                 IF  ( Size  >  Max  )
56                 {
57                     Max  = Size
58                     MaxAddr  = Addr
59                 }
60             }
61         }
62     }
63     IF  ( Max  ==  0  )
64     {
65         RETURN
66     }
67     Addr  = MaxAddr  +  10
68     Maxx  = ID[CamID]: ADDR[Addr(b)]
69     Addr  = Addr  +  1
70     Minx  = ID[CamID]: ADDR[Addr(b)]
71     Addr  = Addr  +  1
72     Maxy  = ID[CamID]: ADDR[Addr(b)]
73     Addr  = Addr  +  1
74     Miny  = ID[CamID]: ADDR[Addr(b)]
75 }
```

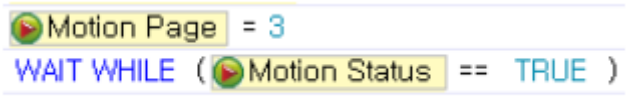
4. Složitější roboti

Roboti, kteří jsou uvedení v této kapitole, potřebují ke svému řízení program, který je složen ze dvou samostatných souborů.

Jeden soubor je tzv. řídicí, s příponou **tsk**. Tento soubor obsahuje logiku chování celého robota. V tomto souboru je kód, který zpracovává údaje ze snímačů, z dálkového ovládače a zjišťuje, zda-li bylo stisknuto nějaké tlačítko na řídicí jednotce. Na základě těchto vnějších podnětů robot vykonává určité pohyby.

Tyto pohyby jsou však uvedeny v samostatném souboru s příponou **mtn**. V tomto souboru je v podstatě uložena tabulka, která obsahuje skupiny jednotlivých pozic. Každá tato skupina má svůj vlastní číselný a textový identifikátor.

Řídicí jednotky typu CM-x však mají možnost, tyto skupiny jednotlivých pozic vykonávat postupně tak, že se tato skupina jeví jako jeden složitý pohyb. Aby však tato skupina pozic mohla být vykonaná, musí být v souboru tsk následující instrukce. Viz. následující obrázek 4.1.



Obr. 4.1 – Vyvolání skupiny jednotlivých pozic

Instrukce na prvním řádku vyvolá vykonávání skupiny pozic, které jsou uloženy v mtn souboru na řádku 3. Avšak tyto pozice jsou vykonávány v jiném prováděcím toku (threads). Což znamená, že od tohoto okamžiku se začne současně vykonávat pohyb a zároveň se bude pokračovat ve zpracovávání tsk souboru.

Vzhledem k tomu, že zpracovávání tsk souboru je principiálně mnohem rychlejší než vykonávání pohybů, tak by toto asynchronní vykonávání mohlo vést k ignorování některých instrukcí, případně i některých pozic (a to v případě, že by v souboru tsk následovala další instrukce, která by odstartovala další řádek z mtn souboru – čímž by se předchozí vykonávání zastavilo a automaticky by se začaly vykonávat nové pohyby).

Toto chování může být někdy žádoucí. Avšak v situacích, kdy chceme synchronně vykonávat činnosti robota, je nutné za první řádek kódu ještě doplnit druhý, tak jak je vidět na obrázku 4.1.

Tj. tento druhý řádek má takový význam, že se na tomto místě čeká tak dlouho, dokud se točí nějaký pohon. Tzn. Motion Status je proměnná řídicí jednotky, která sleduje jednotlivé stavové bity pohonných jednotek a pokud jeden z pohonných jednotek je v pohybu, tak tato proměnná má logickou 1.

V tomto případě se jedná o kód, který způsobí čekání zpracovávání tsk souboru. To může způsobit nereagování na vnější podněty v případě příliž dlouhých pohybových skupin. Proto je nutné na počátku určit, co vlastně chceme, aby robot vykonával a jak má reagovat.

4.1. Robot Humanoid – typ A

PRAKTICKÁ ÚLOHA

Tento robot je vytvořen z celkem 18-ti pohonů AX-12A, kde chůzi zajišťuje celkem 12 pohonů. Ostatní pohony jsou použity na paže. Z tohoto důvodu robot nehýbe hlavou a nedokáže se otáčet v pase. Pokud by čtenář tuto možnost vyžadoval, může tohoto robota modifikovat a zbývající pohony doplnit dle své potřeby. Touto úpravou dojde ke změně těžiště robota a veškeré připravené řídicí algoritmy nebudou použitelné, v lepším případě bude možné použít jen fragmenty.

Tento robot obsahuje celkem dva hlavní snímače, které je možné využit k interakci s okolím. Na hrudi robota je umístěn optický snímač vzdálenosti překážek, který je citlivý v rozsahu cca 5 až 30 cm. Druhý snímač je elektronický gyroskop, který je umístěn uvnitř těla robota (pod řídicí jednotkou). Tento snímač je velice důležitý pro algoritmy simulující chůzi. Díky tomuto snímači je možné, aby se robot pohyboval po různých materiálech, jako např. linoleum, koberec s nízkým vlasem, beton, kachličky, apod.

Vzhledem ke konstrukci chodidla se robot nedokáže pohybovat po kamenitém či zvlněném povrchu. Pokud se robot pohybuje a během chůze upadne tak, že zůstane ležet na břiše nebo na zádech, pak tento snímač je možné využít k detekci pádu. Pozor! Není možné detekovat pozici, ale pouze pád. Toto omezení vychází z principu použitého snímače, tzn. elektronický gyroskop, který je zde použit, reaguje pouze na úhlovou rychlost nezávisle, ve dvou osách.

ČAS K SESTAVENÍ: 720 minut

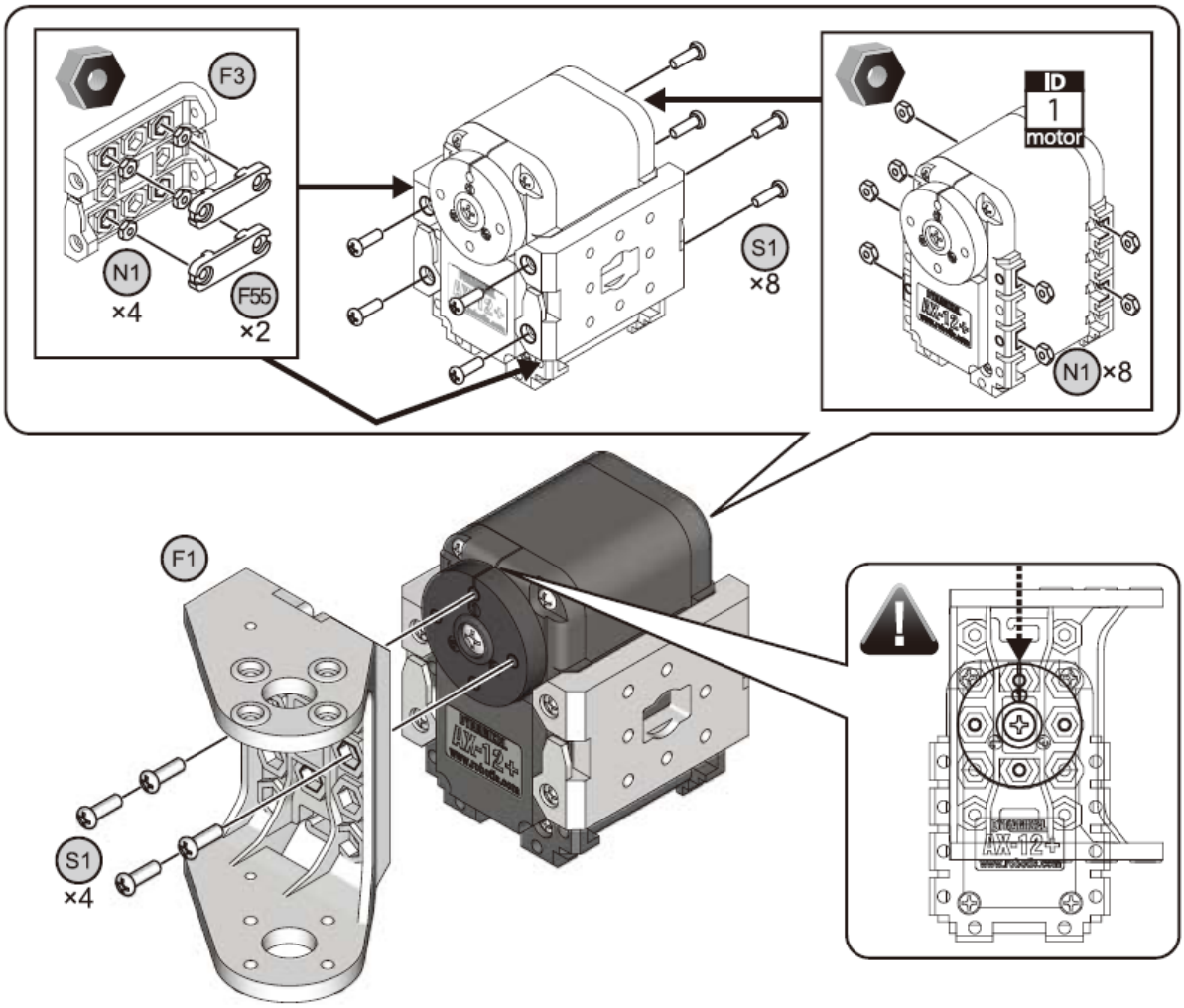
CÍL

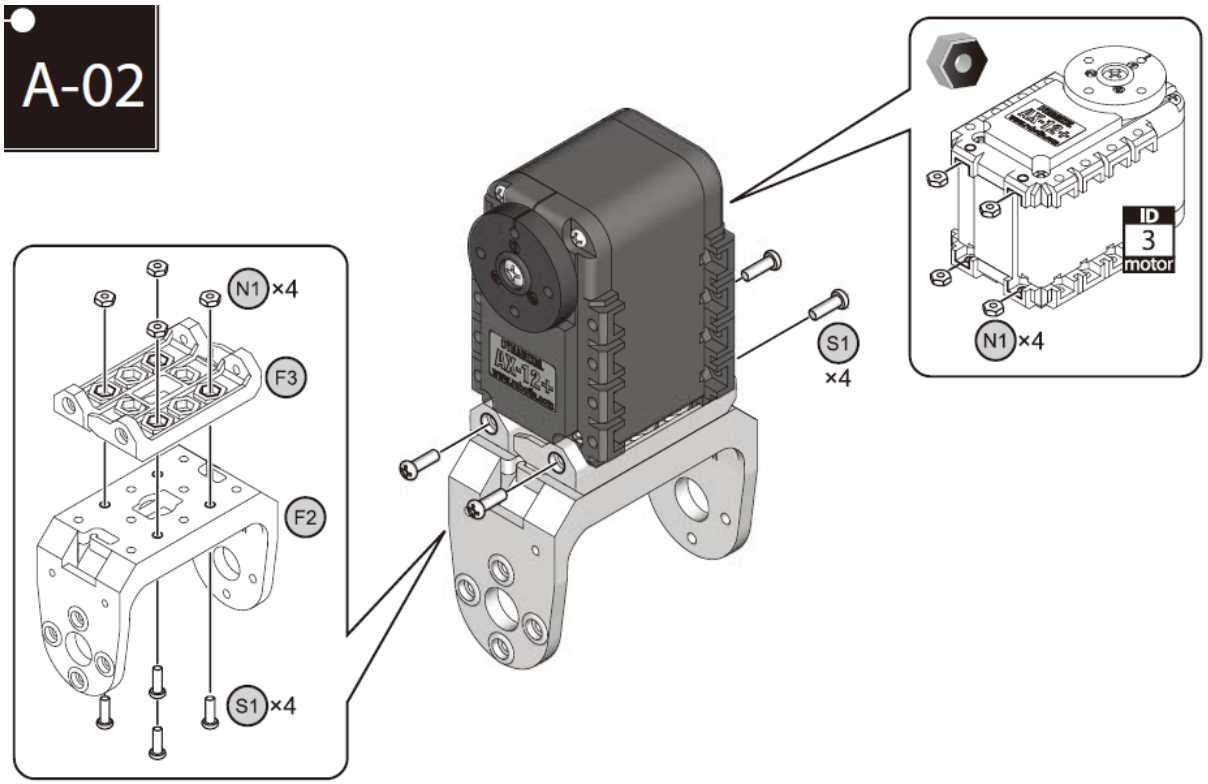
Sestavením tohoto robota získá čtenář možnost si prakticky ověřit řídicí algoritmy humanoidního robota. Tyto algoritmy jsou zaměřeny na ukázku chůze, otáčení kolem své vlastní osy a některých dalších specifických pohybů tělem a pažemi.

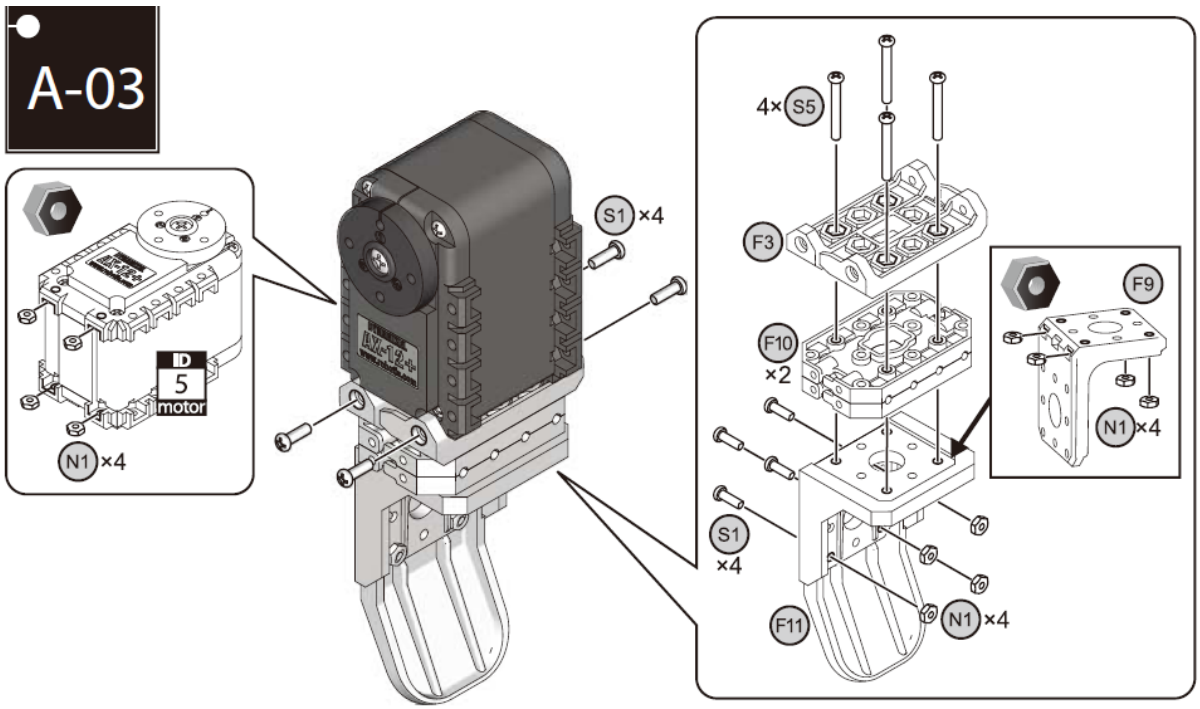
Postup sestavení robota:

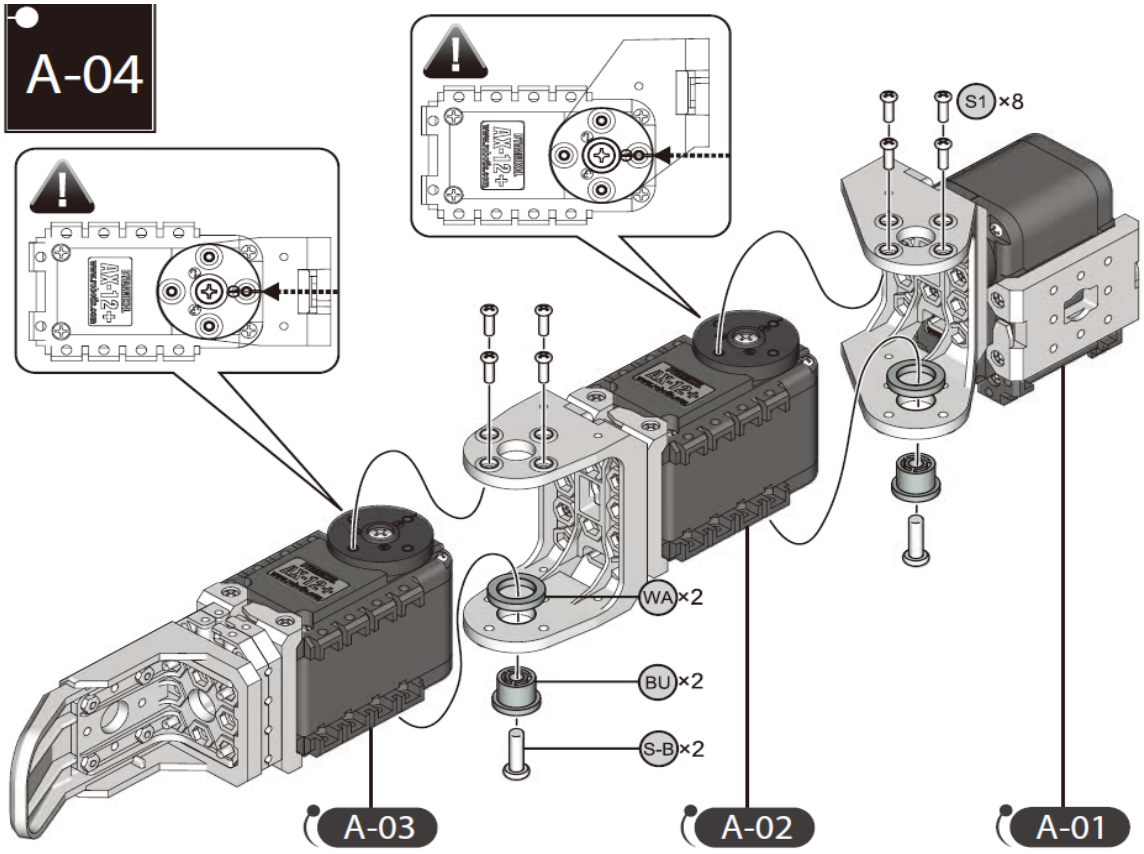
Protože se jedná o Humanoida typu A, budou následující kroky označeny A-01, A-02, A-03...

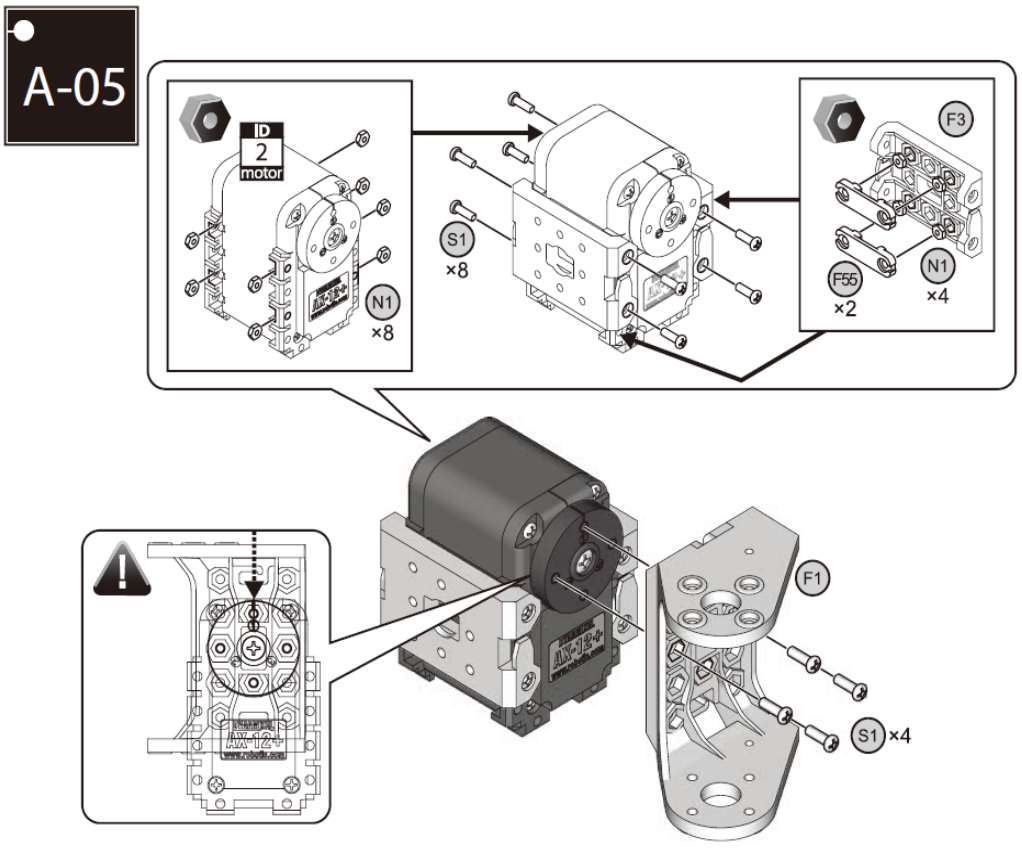
A-01

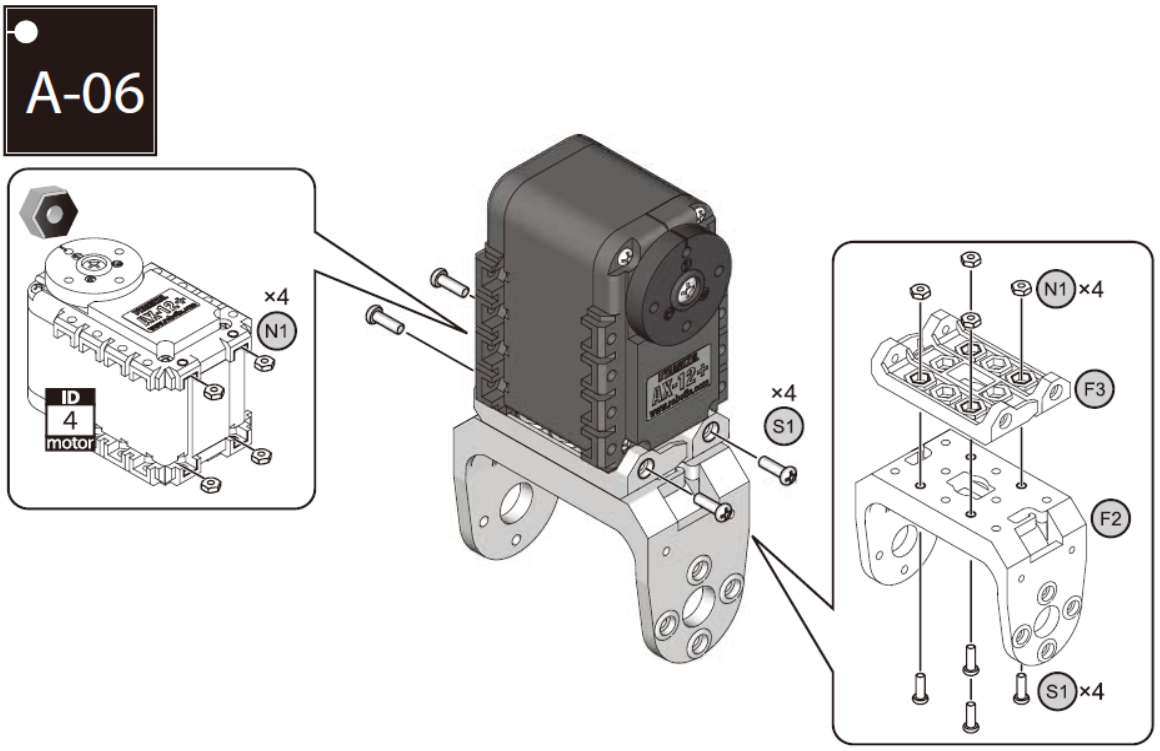


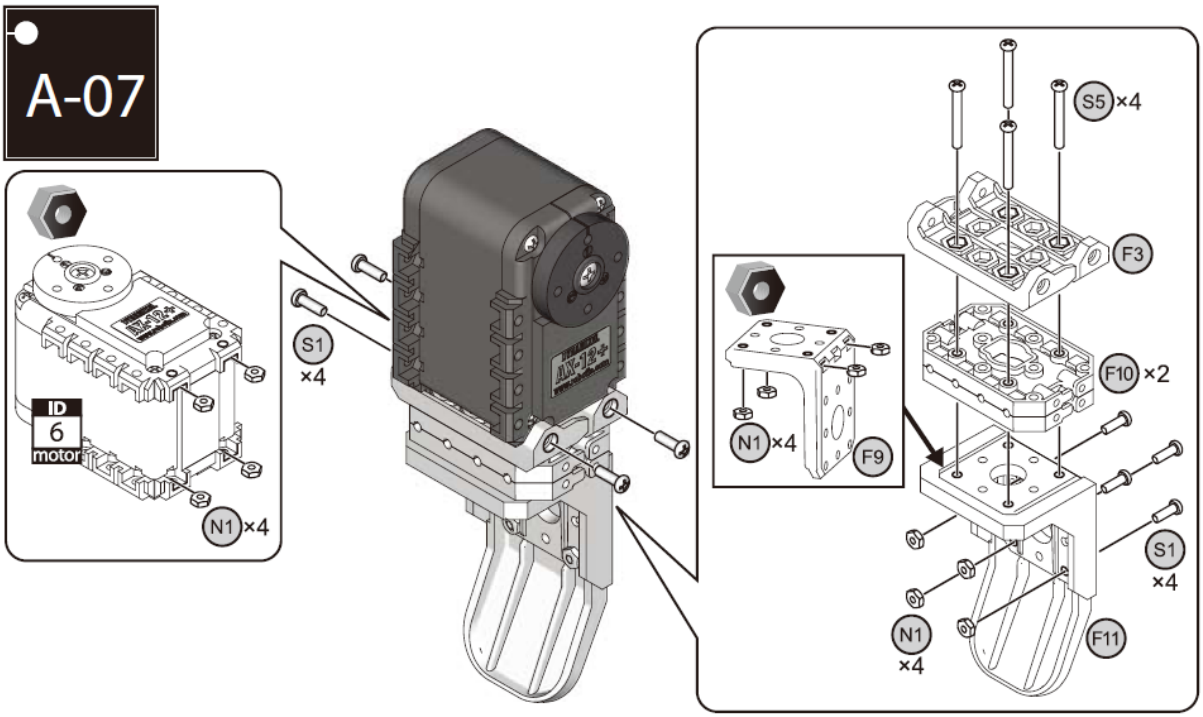

A-02
N1 ×4
F3
F2
S1 ×4
S1 ×4
ID 3 motor
N1 ×4

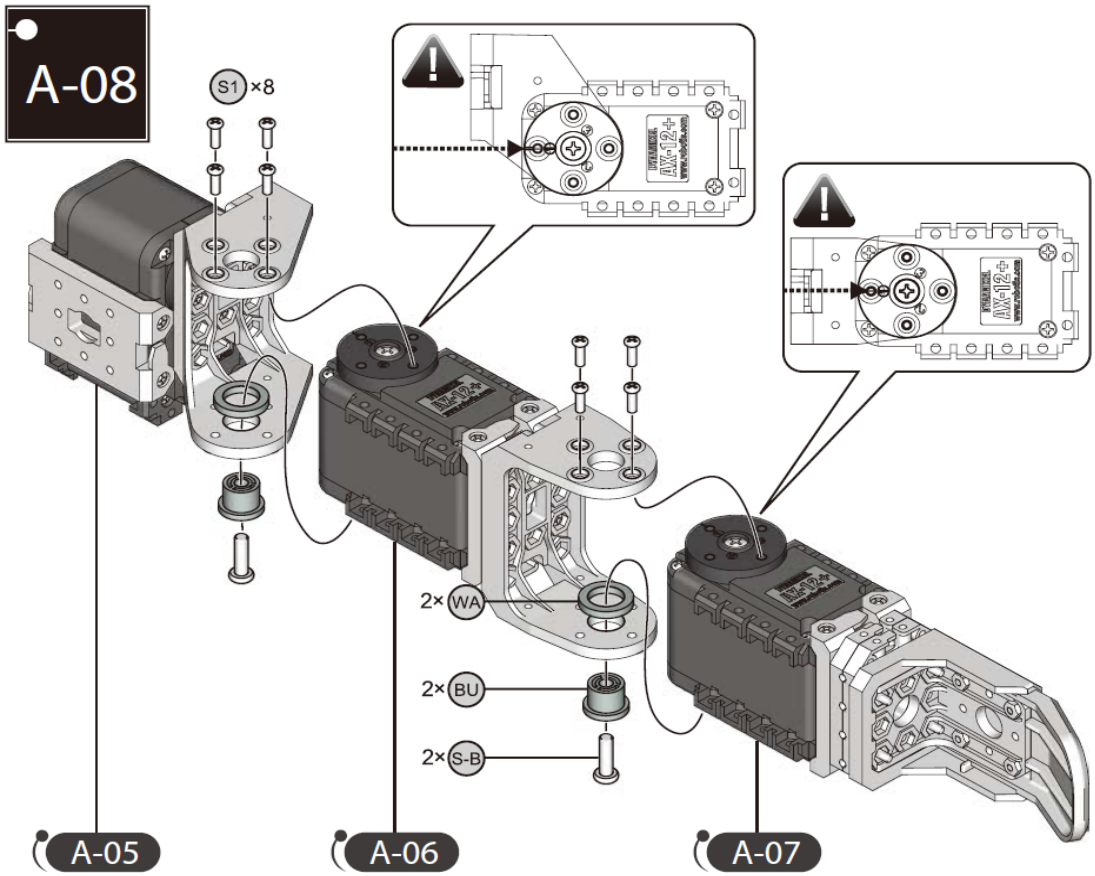

A-03
ID 5 motor
N1 ×4
S1 ×4
4× S5
F3
F10 ×2
F9
N1 ×4
S1 ×4
F11
N1 ×4

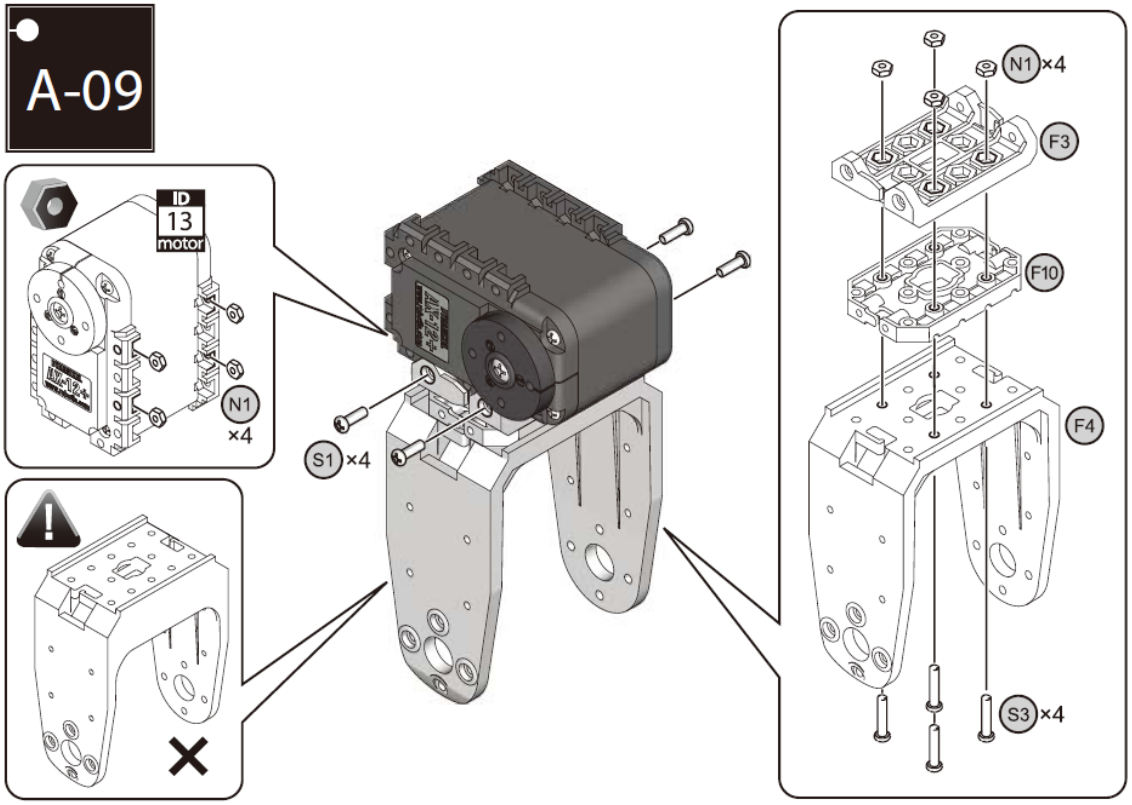

A-04
S1 ×8
WA ×2
BU ×2
S-B ×2
A-03
A-02
A-01

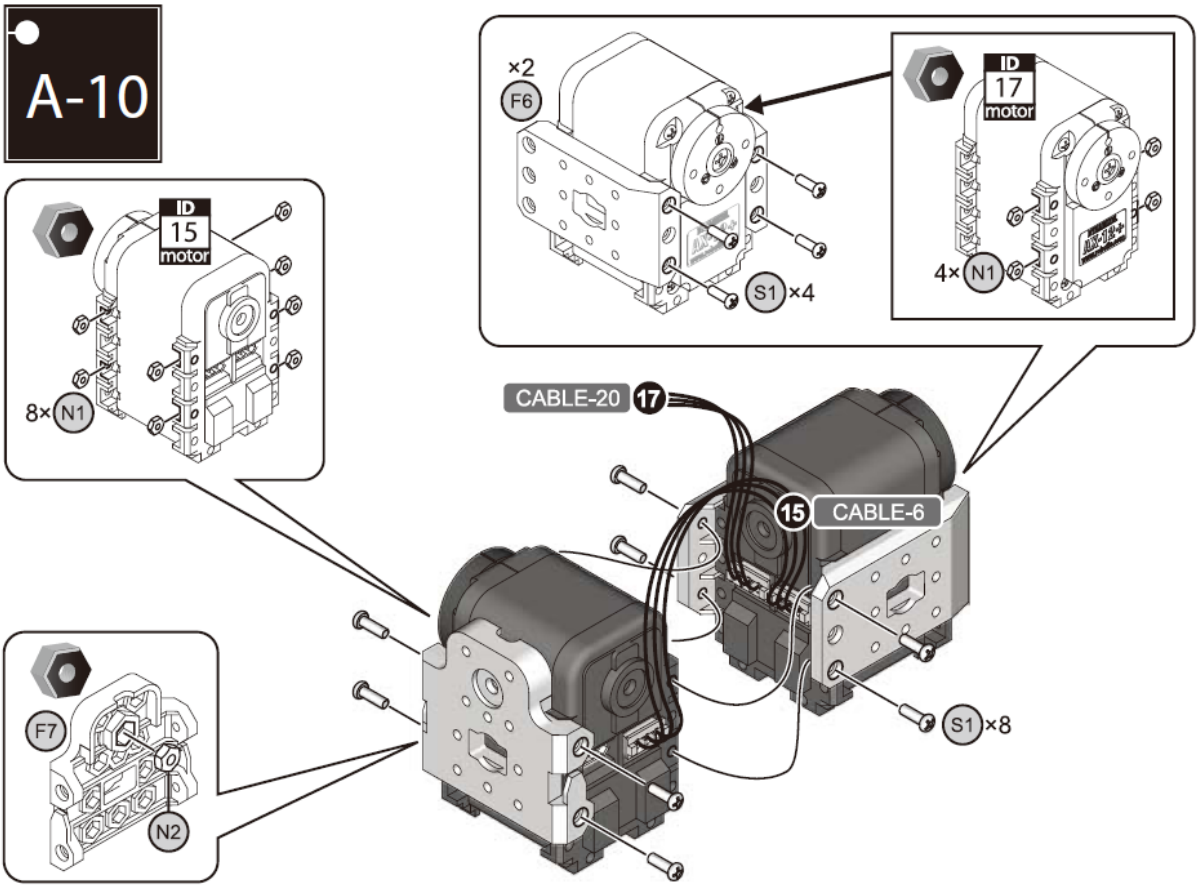

A-05
ID 2 motor
N1 ×8
S1 ×8
F3
F55 ×2
N1 ×4
F1
S1 ×4

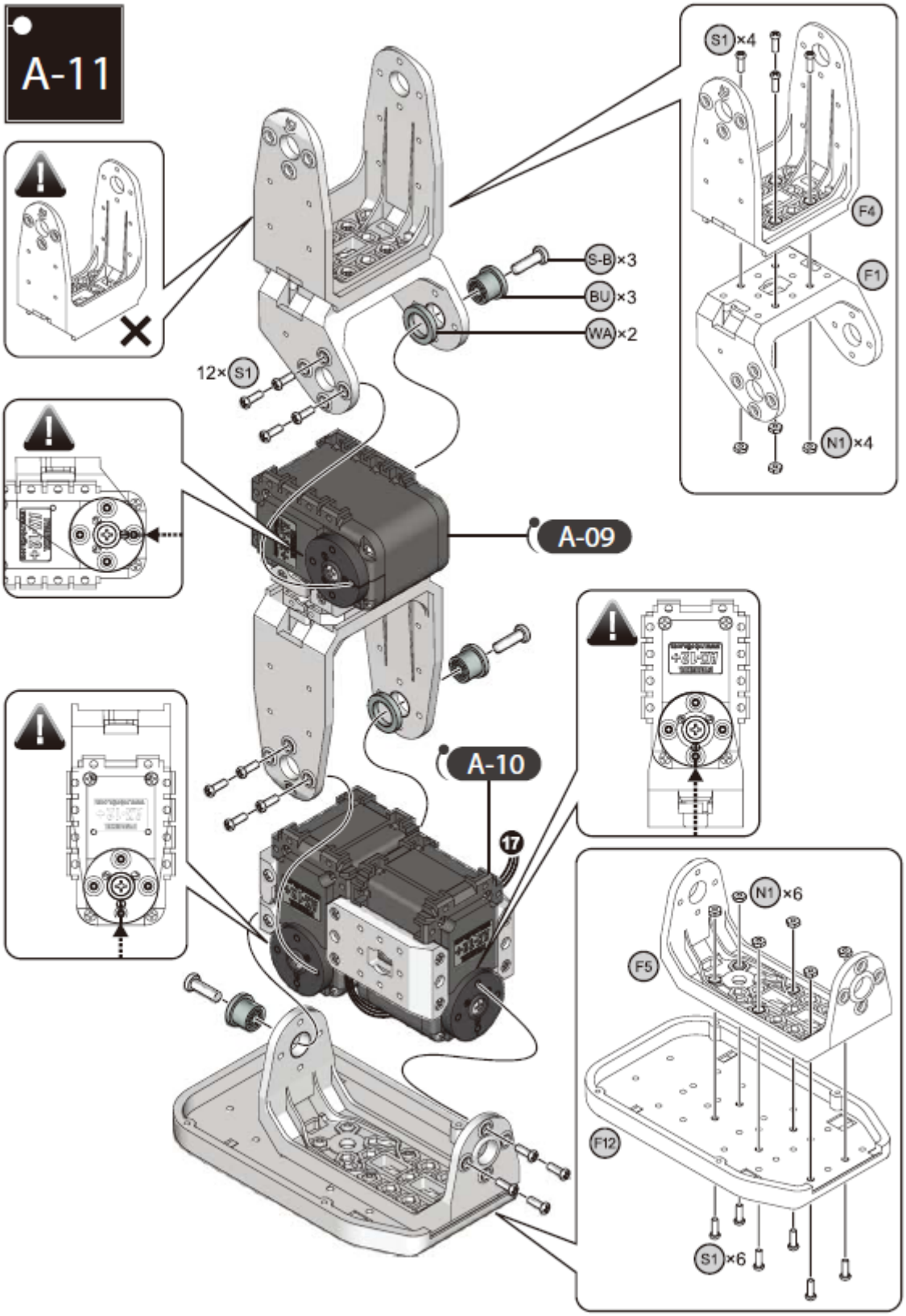

A-06
×4
N1
ID
4
motor
×4
S1
N1 ×4
F3
F2
S1 ×4

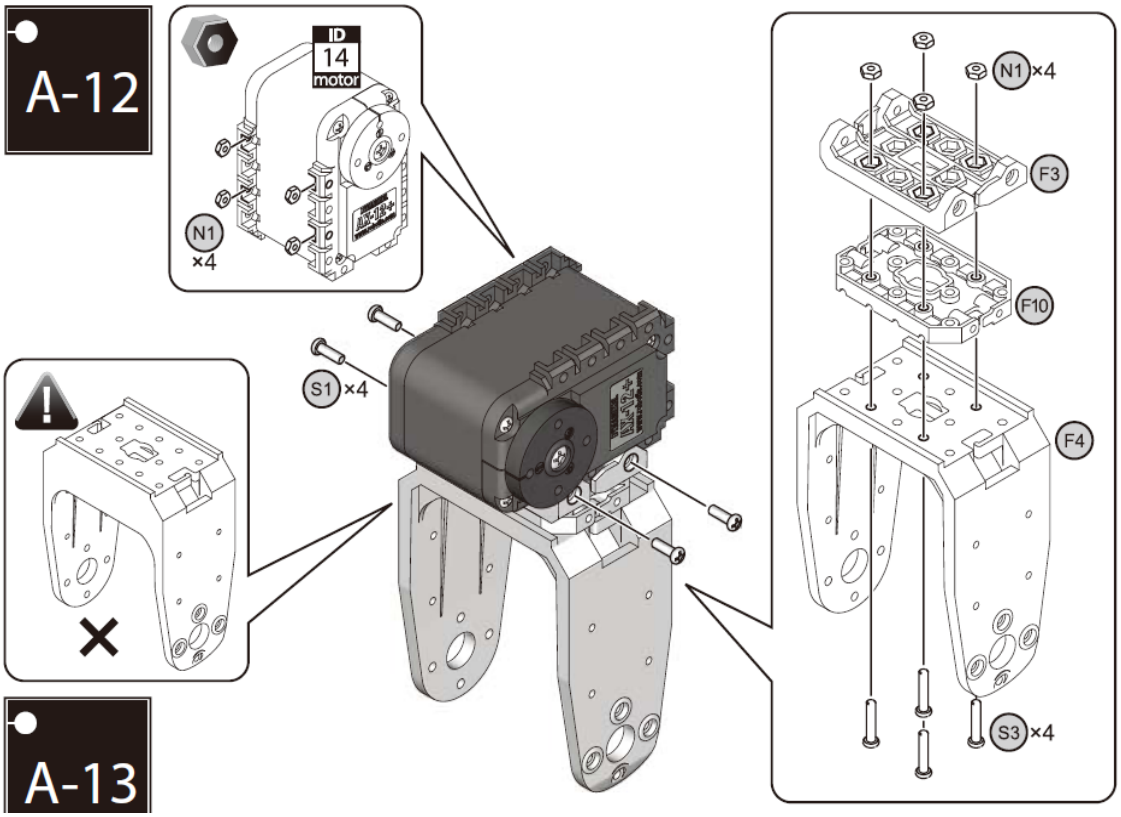

A-07
ID
6
motor
N1 ×4
S1
×4
S5 ×4
F3
F10 ×2
N1 ×4
F9
S1
×4
N1
×4
F11

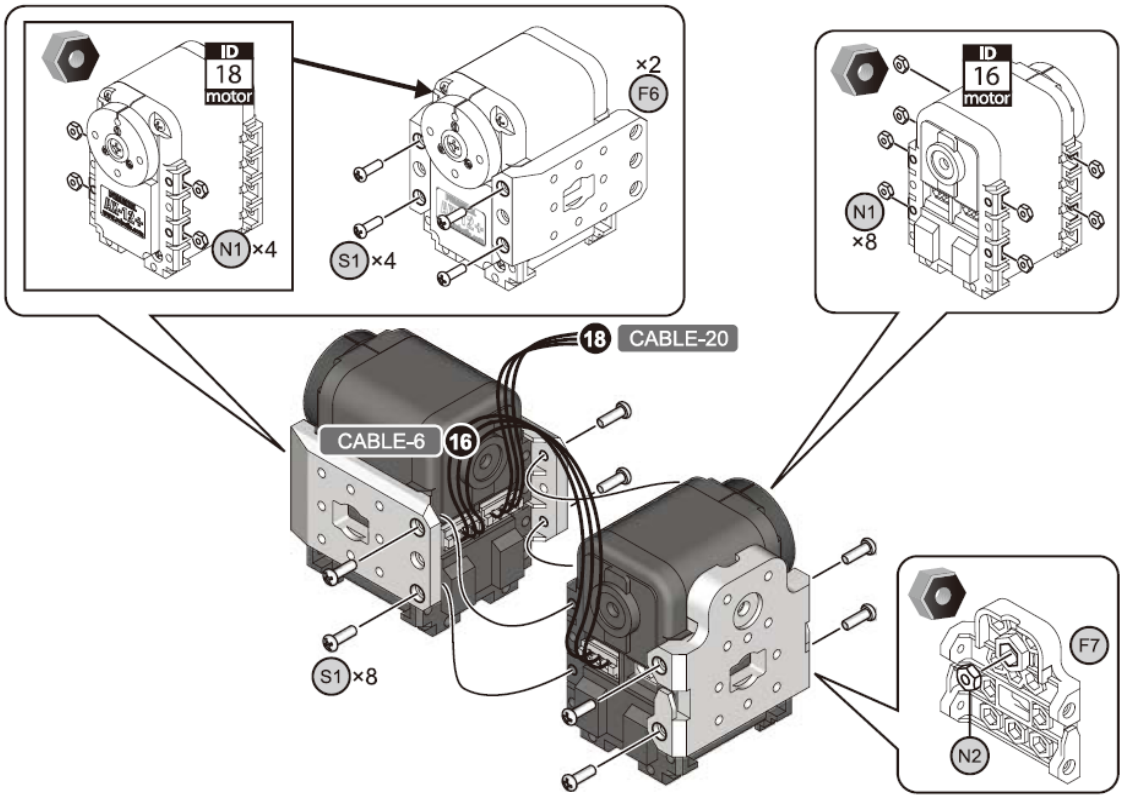

A-08
S1 ×8
2× WA
2× BU
2× S-B
A-05
A-06
A-07

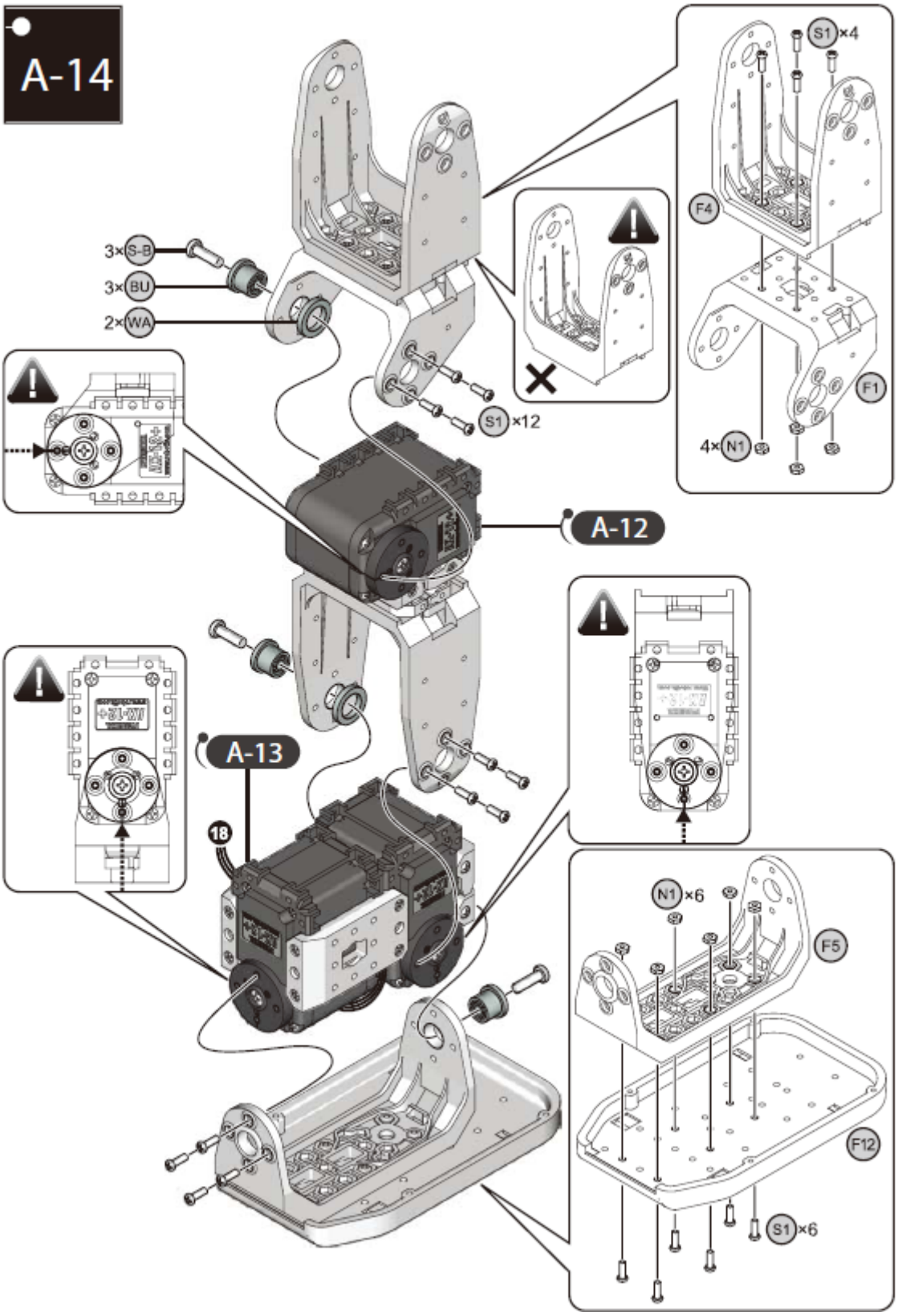

A-09
ID 13 motor
N1 ×4
S1 ×4
N1 ×4
F3
F10
F4
S3 ×4


A-10
ID
15
motor
8× N1
×2
F6
S1 ×4
ID
17
motor
4× N1
CABLE-20
17
15
CABLE-6
S1 ×8
F7
N2

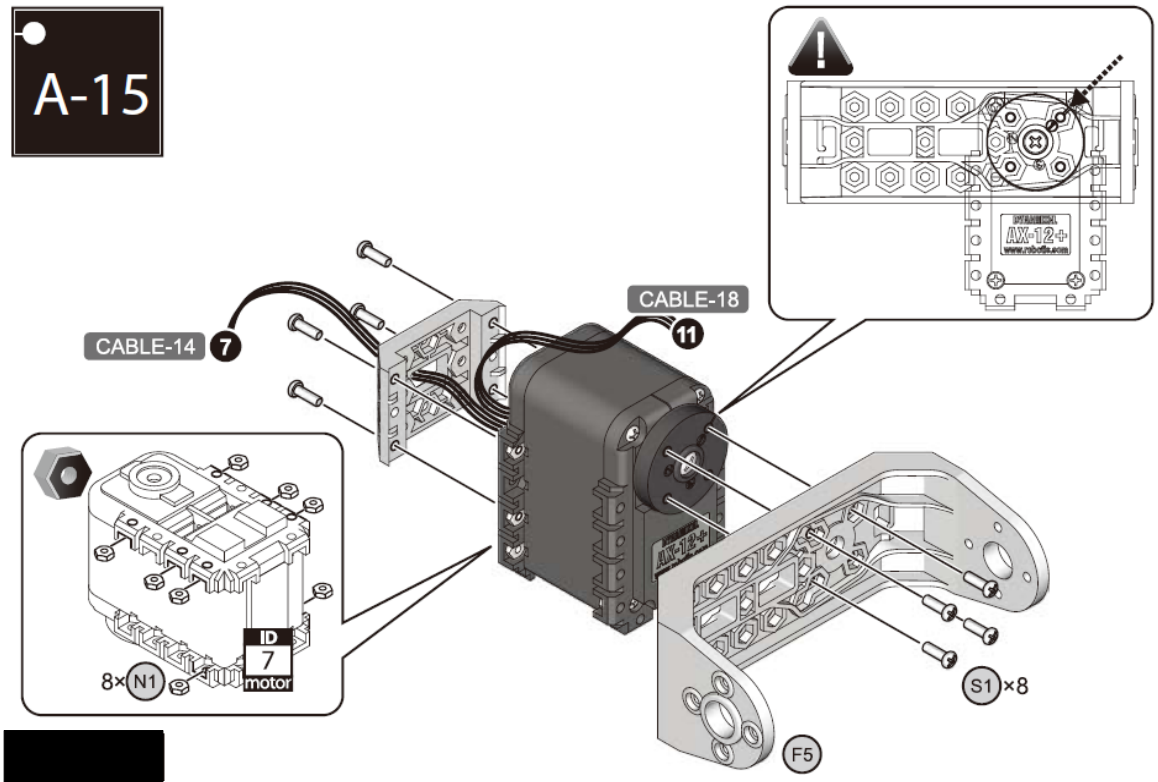

A-11
S1 ×4
F4
F1
N1 ×4
S-B ×3
BU ×3
WA ×2
12× S1
A-09
A-10
17
N1 ×6
F5
F12
S1 ×6


A-12
ID
14
motor
N1
×4
S1
×4
N1
×4
F3
F10
F4
S3
×4
A-13

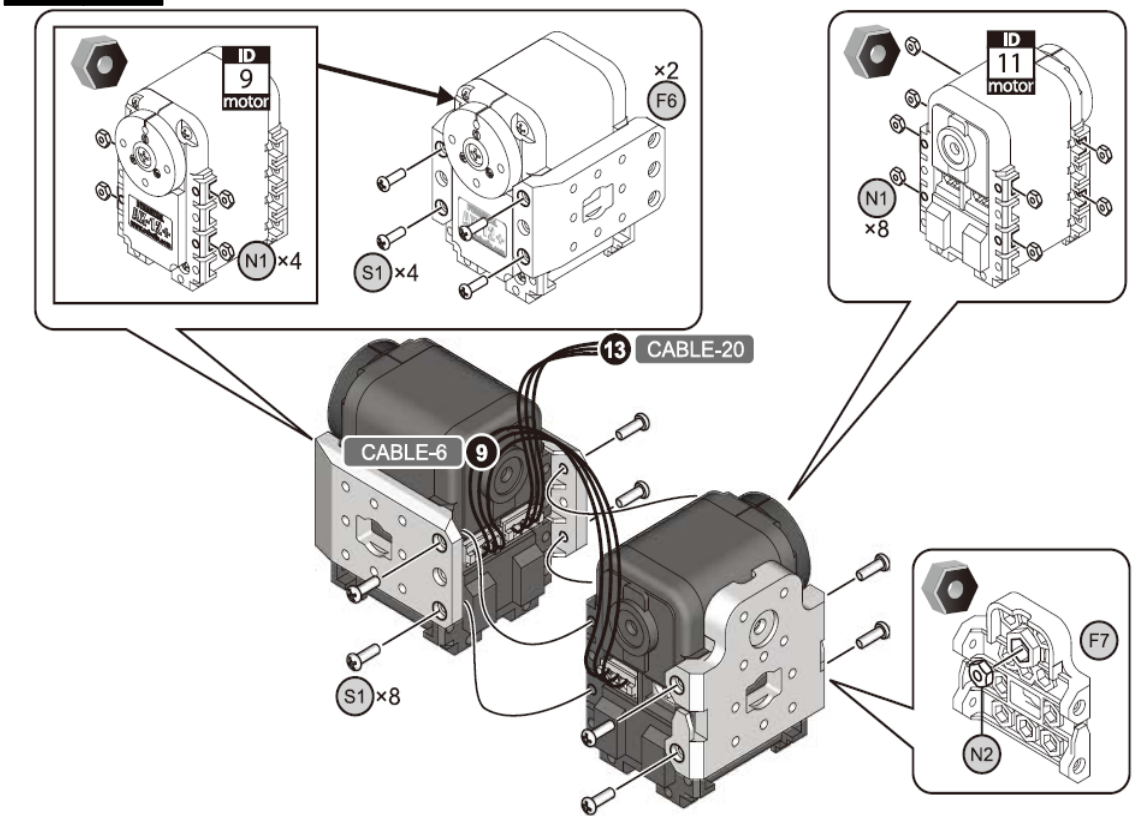

ID
18
motor
N1
×4
×2
F6
S1
×4
ID
16
motor
N1
×8
18
CABLE-20
CABLE-6
16
S1
×8
F7
N2

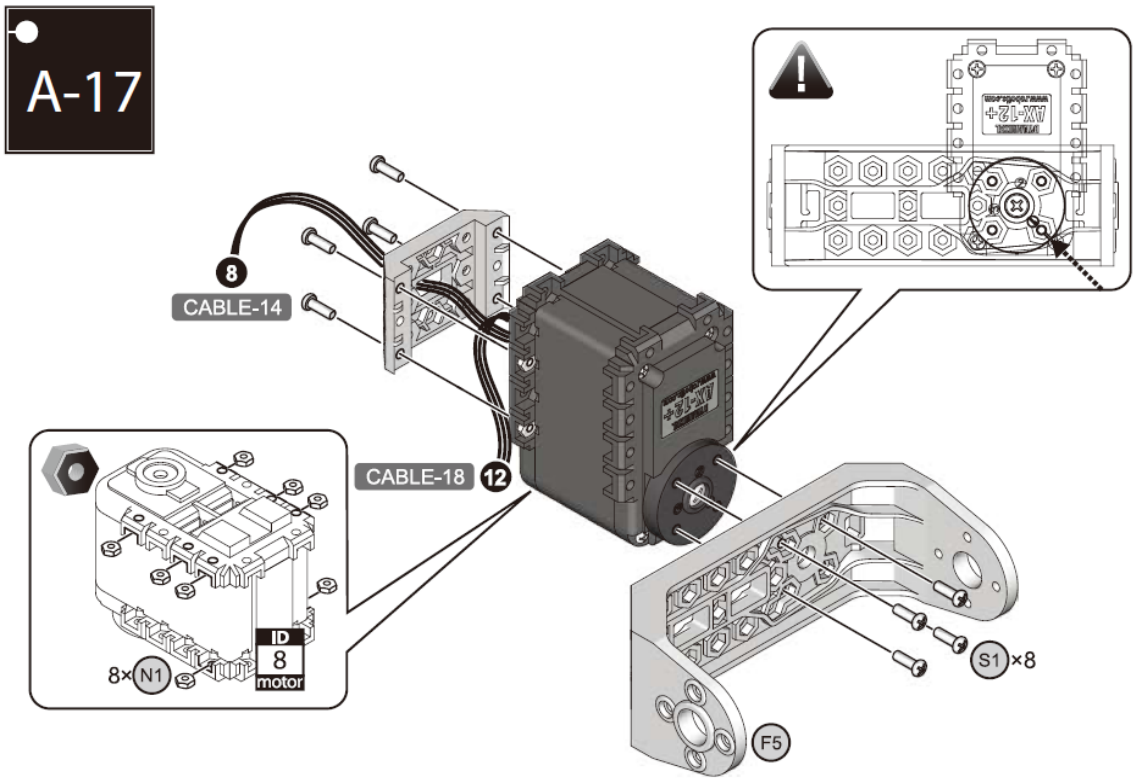

A-14
3×S-B
3×BU
2×WA
S1×12
S1×4
F4
F1
4×N1
A-12
A-13
18
N1×6
F5
F12
S1×6

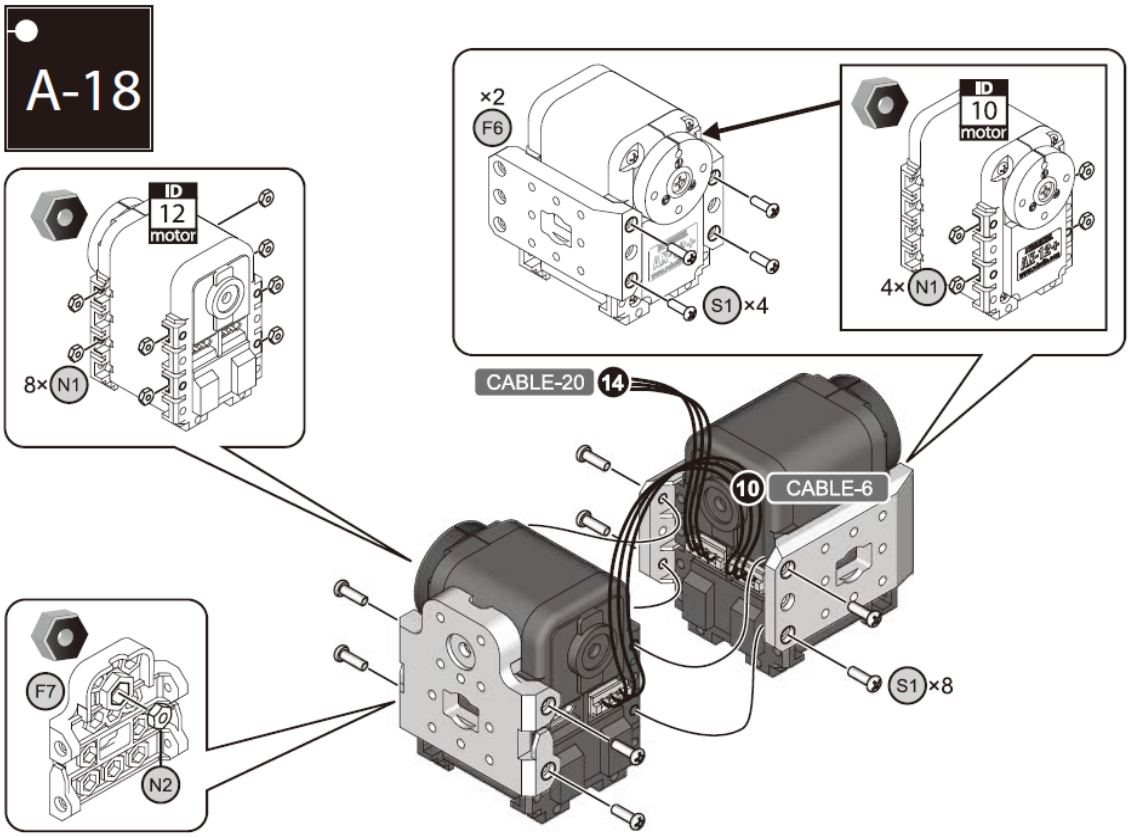
A-15


CABLE-14
7
CABLE-18
11
ID
7
motor
8×N1
S1×8
F5

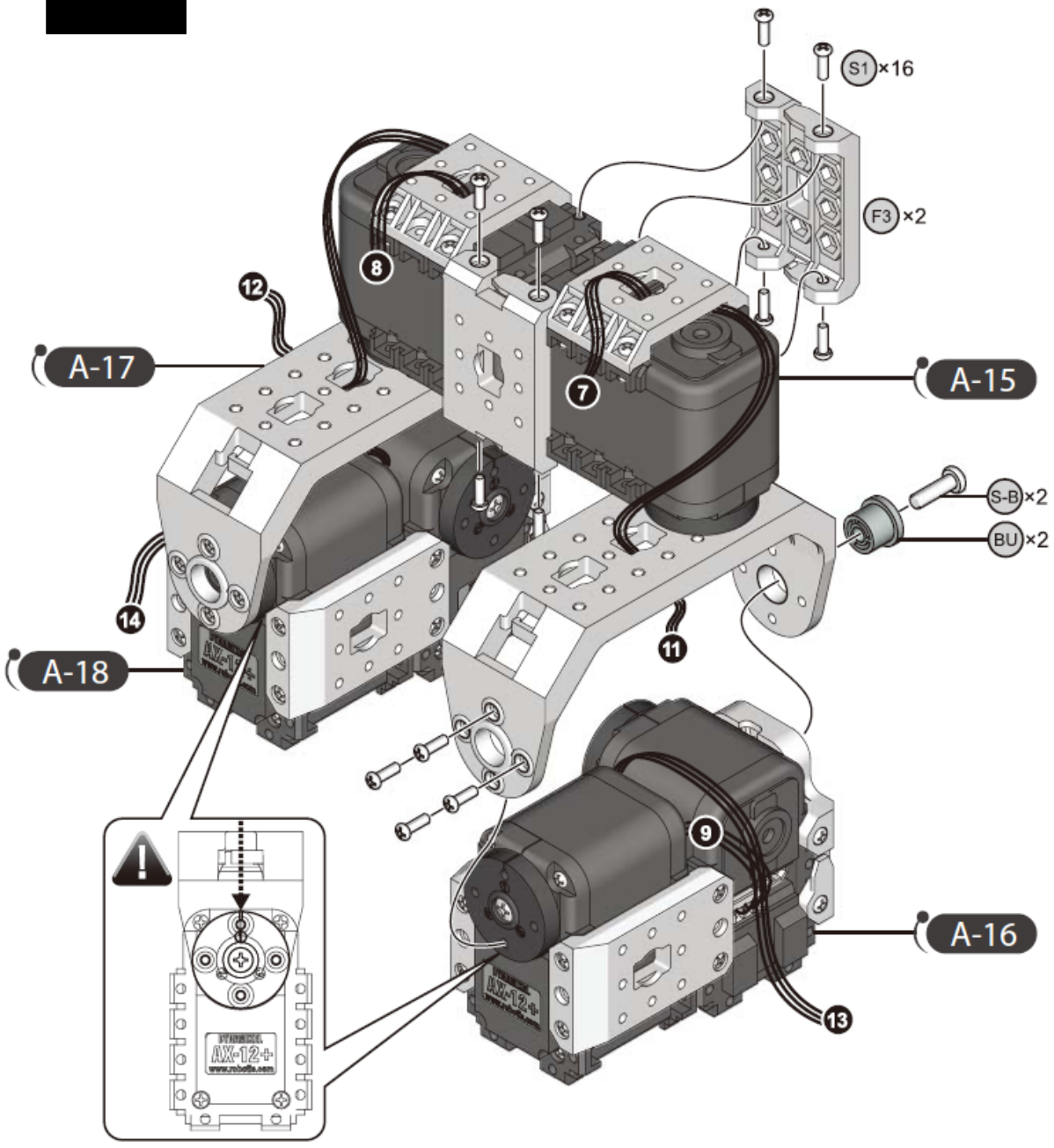
A-16

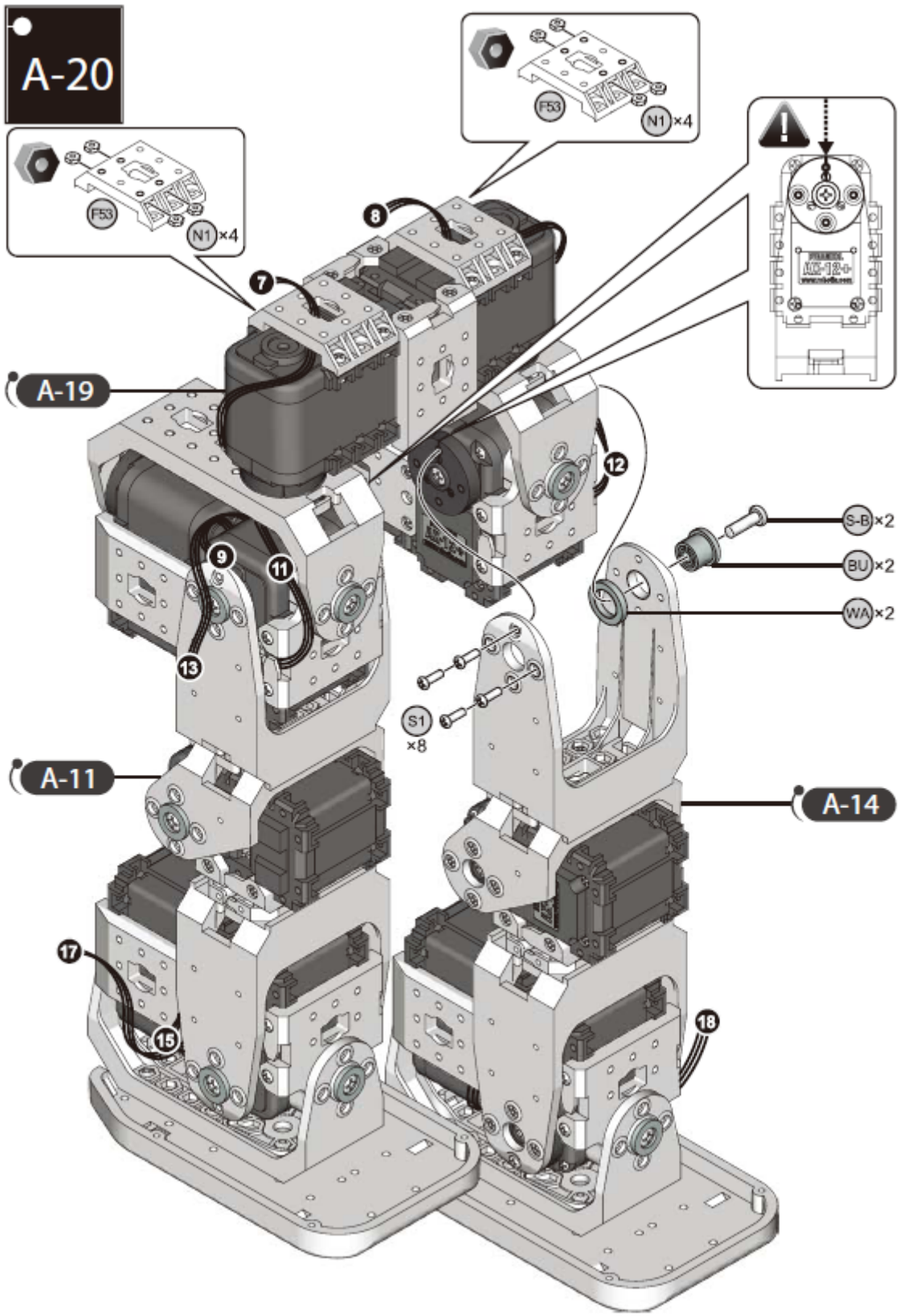

ID
9
motor
N1×4
×2
F6
S1×4
ID
11
motor
N1
×8
13
CABLE-20
CABLE-6
9
S1×8
F7
N2

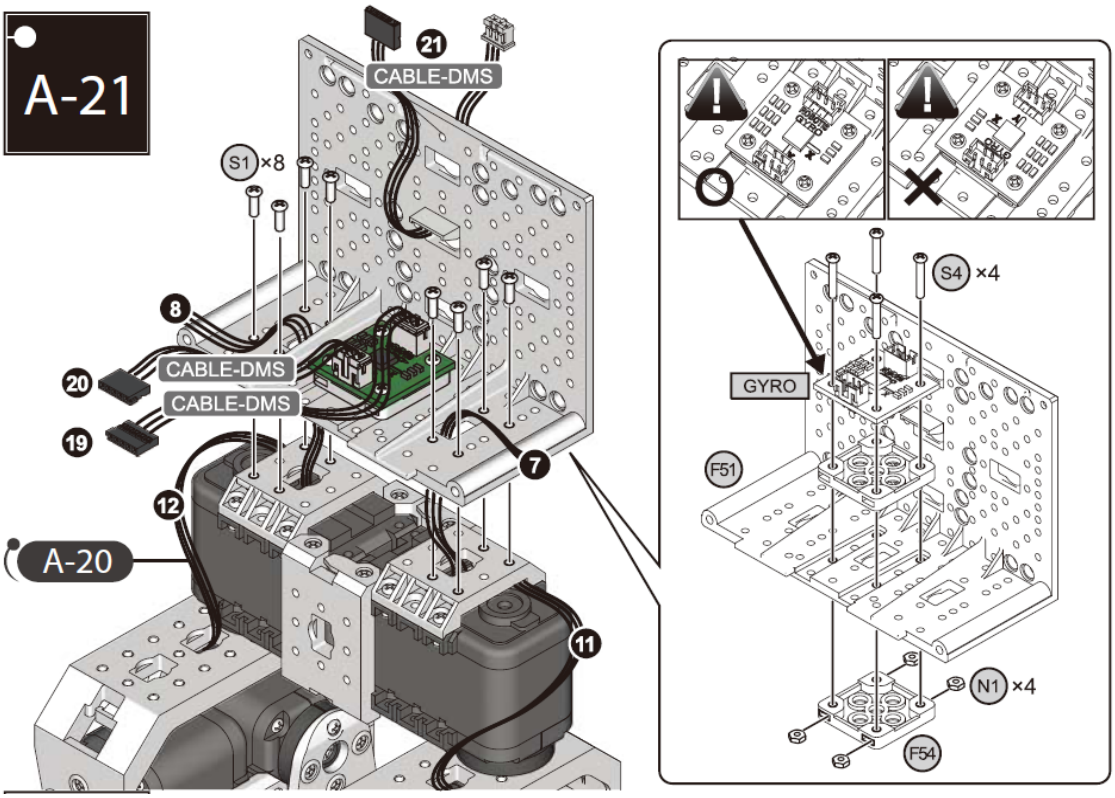

A-17
8
CABLE-14
CABLE-18
12
ID
8
motor
8× N1
S1 ×8
F5

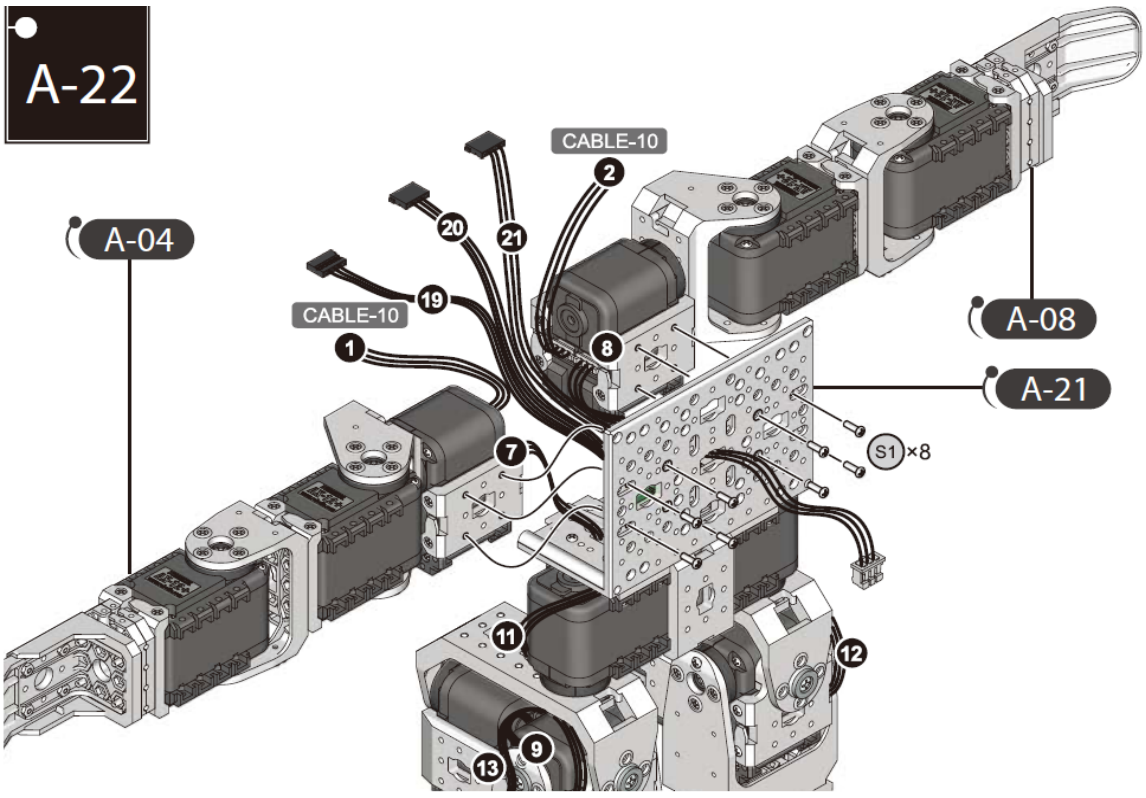

A-18
ID
12
motor
8× N1
×2
F6
S1 ×4
ID
10
motor
4× N1
CABLE-20
14
10
CABLE-6
S1 ×8
F7
N2

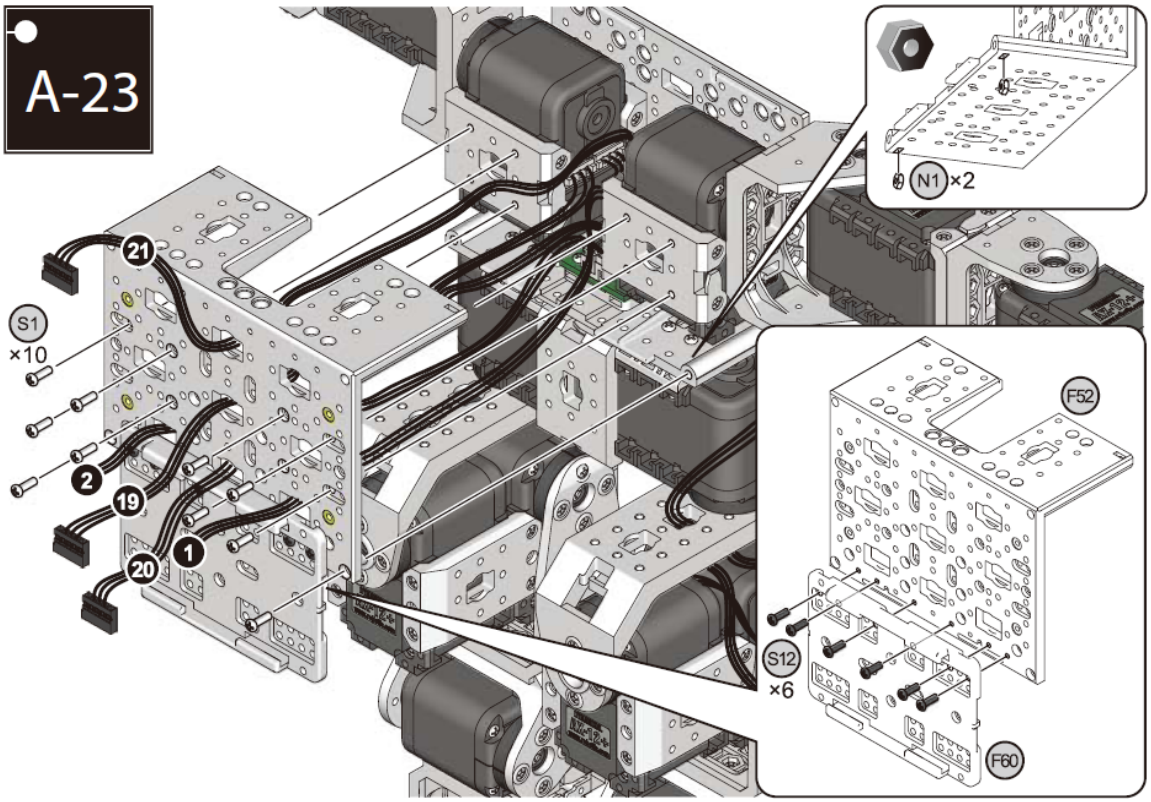
A-19

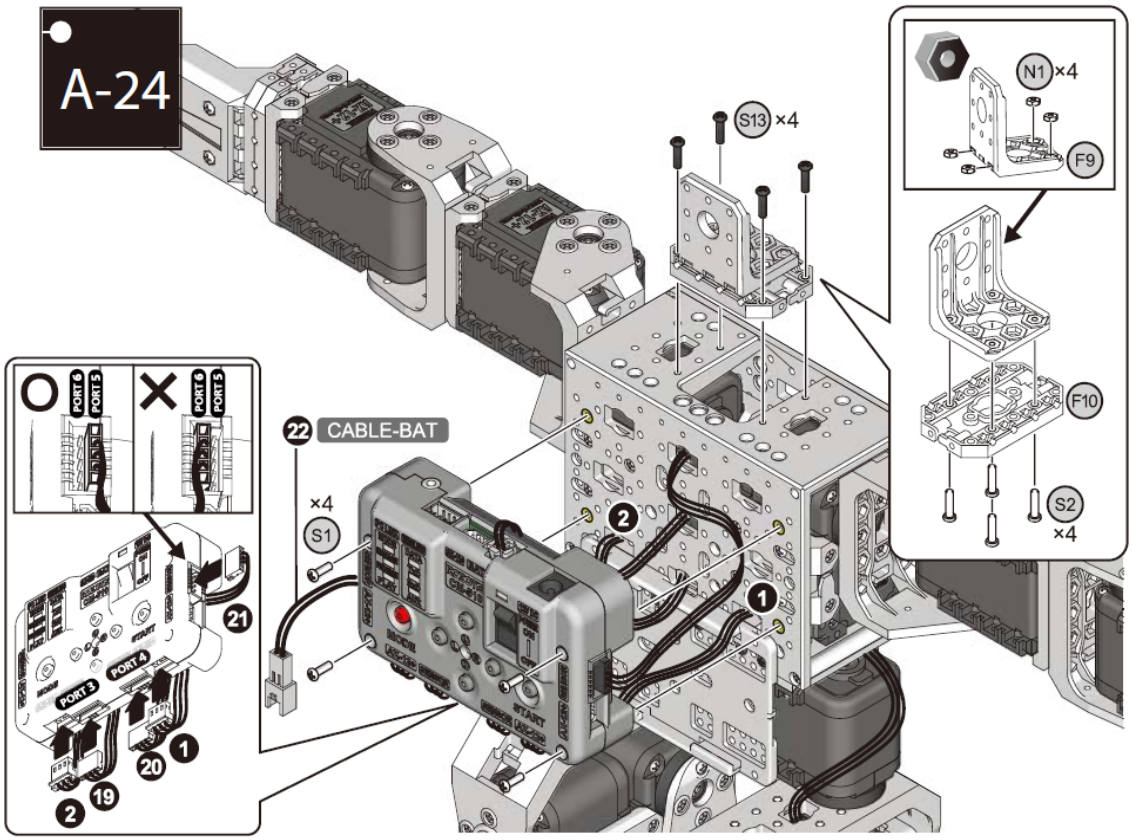



A-20
F53
N1 ×4
F53
N1 ×4
A-19
A-11
A-14
S-B ×2
BU ×2
WA ×2
S1
×8
7
8
9
11
12
13
15
17
18

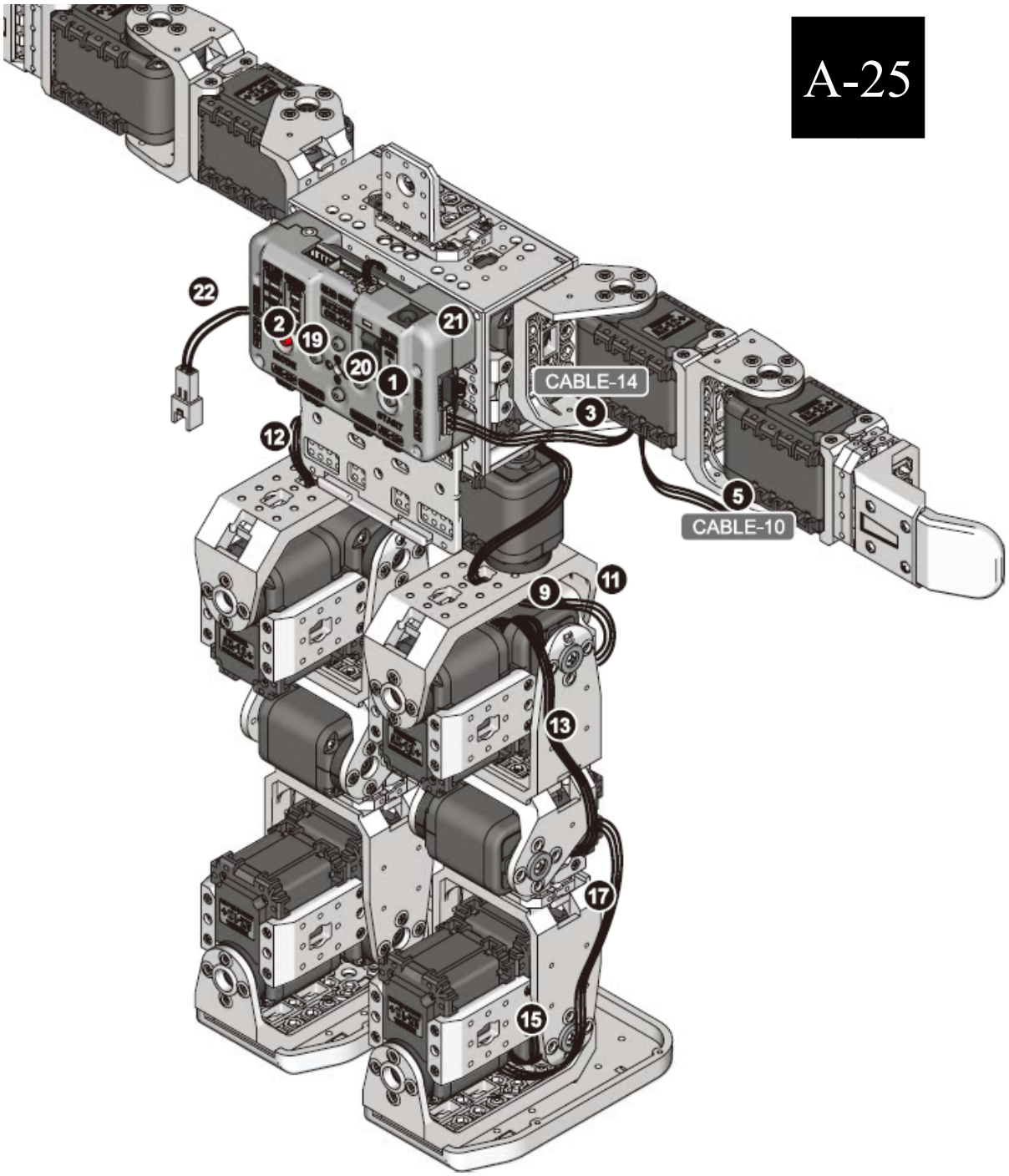

A-21
21
CABLE-DMS
S1 ×8
8
20
CABLE-DMS
CABLE-DMS
19
7
12
A-20
11
S4 ×4
GYRO
F51
N1 ×4
F54


A-22
CABLE-10
2
20
21
A-04
19
CABLE-10
1
8
A-08
A-21
S1 ×8
7
11
12
13
9

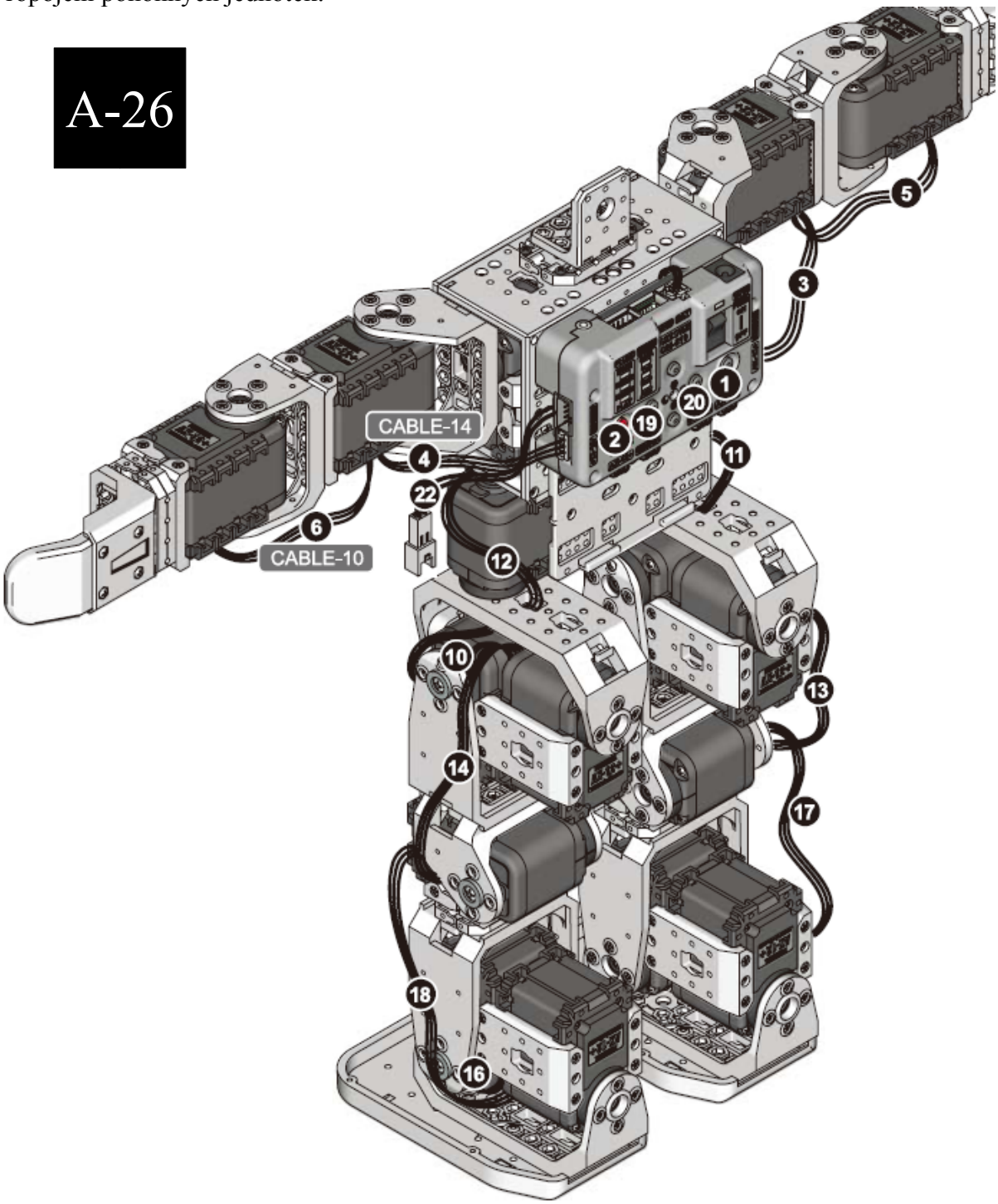

A-23
N1 ×2
S1 ×10
21
2
19
20
1
F52
S12 ×6
F60

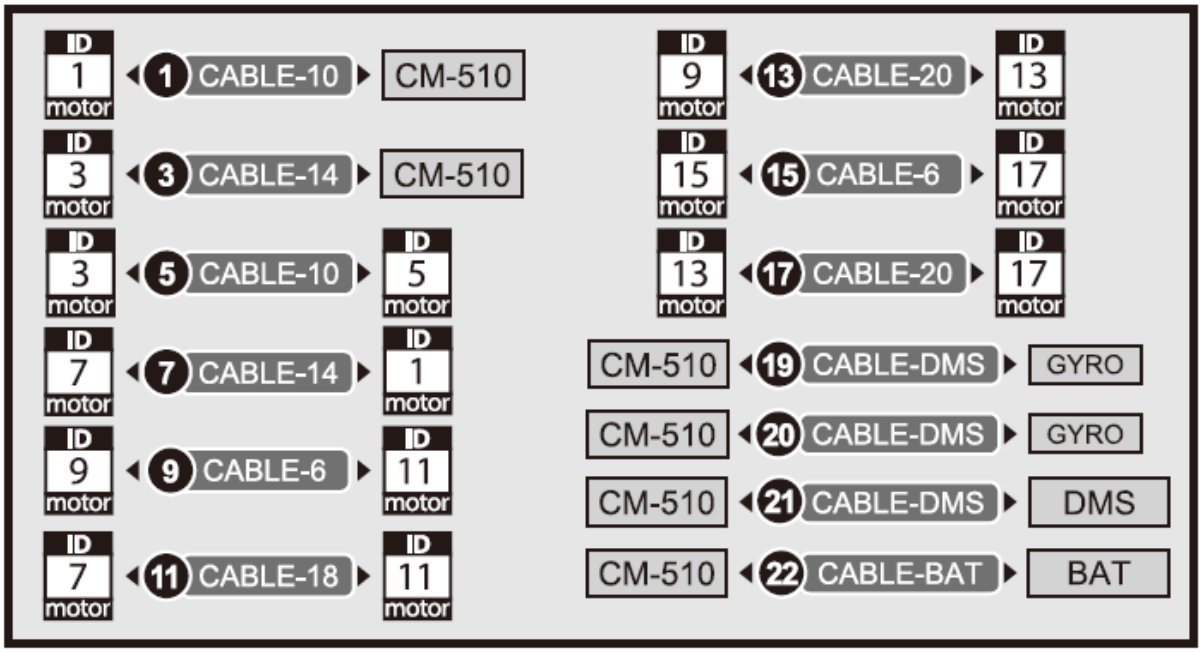

A-24
S13 ×4
N1 ×4
F9
F10
S2 ×4
22 CABLE-BAT
×4
S1
2
1
21
20
19
PORT 1
PORT 2
PORT 3
PORT 4
START

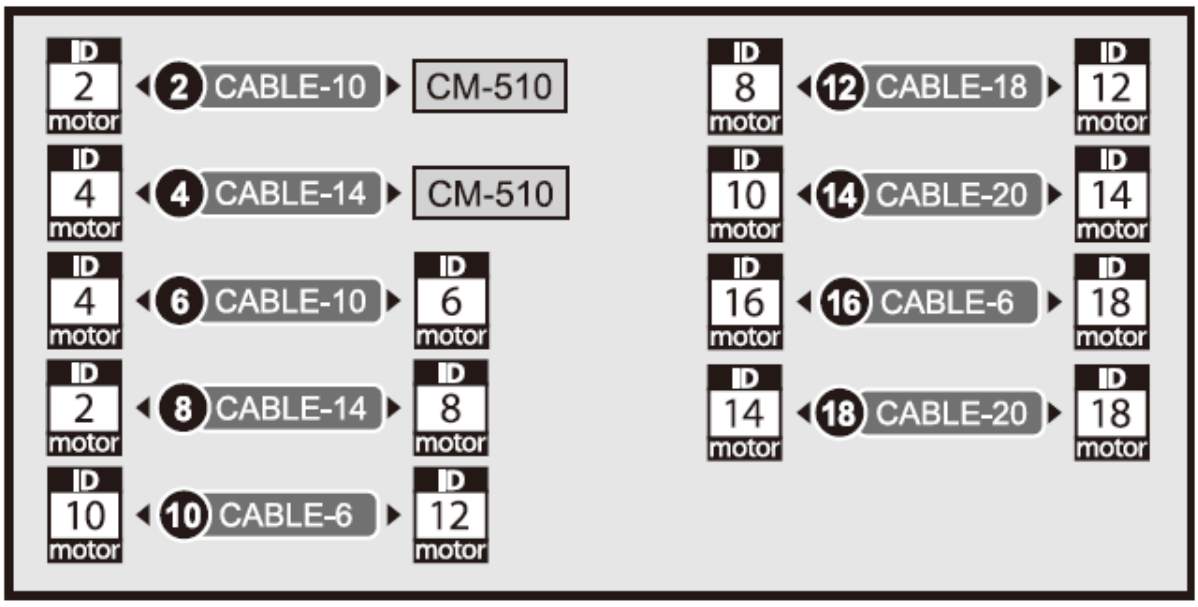
Propojení pohonných jednotek:



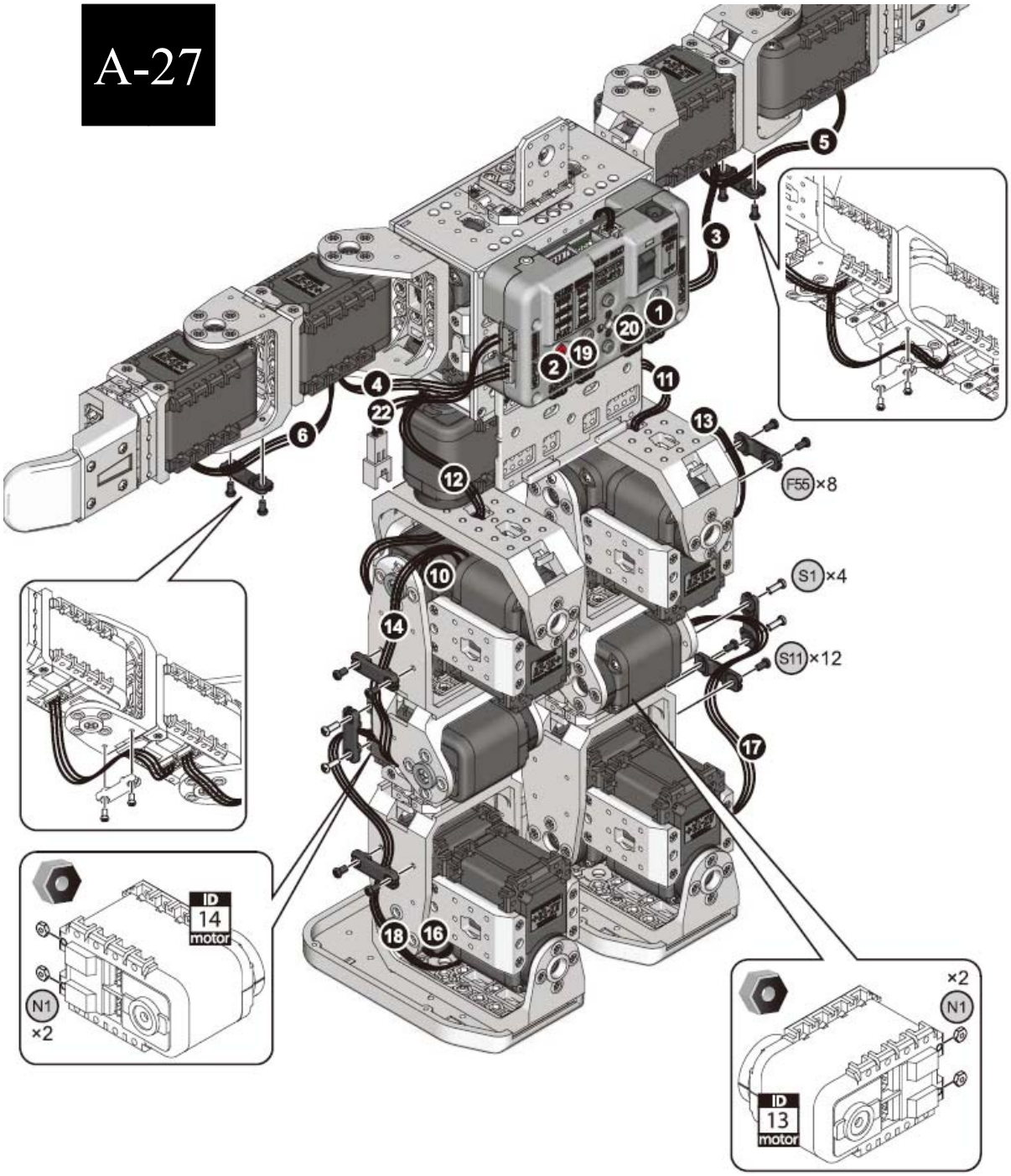
Propojení pohonných jednotek:

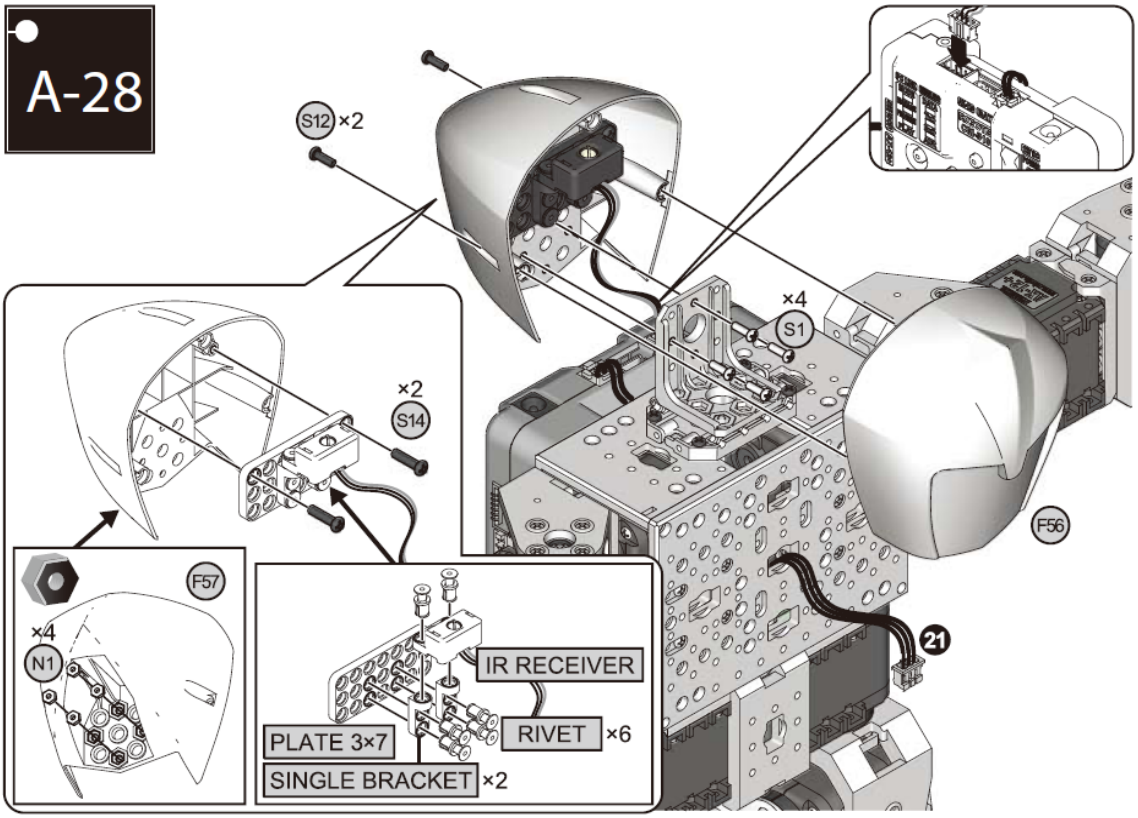


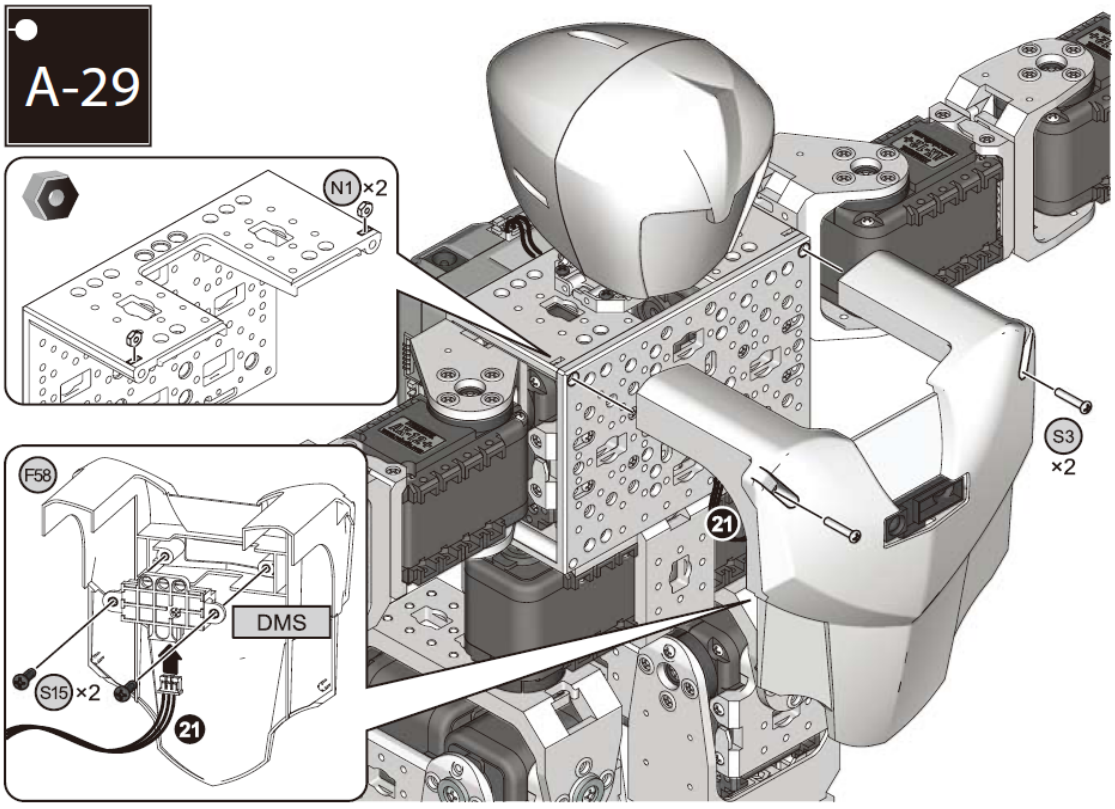

ID 1 motor
1 CABLE-10
CM-510
ID 3 motor
3 CABLE-14
CM-510
ID 3 motor
5 CABLE-10
ID 5 motor
ID 7 motor
7 CABLE-14
ID 1 motor
ID 9 motor
9 CABLE-6
ID 11 motor
ID 7 motor
11 CABLE-18
ID 11 motor
ID 9 motor
13 CABLE-20
ID 13 motor
ID 15 motor
15 CABLE-6
ID 17 motor
ID 13 motor
17 CABLE-20
ID 17 motor
CM-510
19 CABLE-DMS
GYRO
CM-510
20 CABLE-DMS
GYRO
CM-510
21 CABLE-DMS
DMS
CM-510
22 CABLE-BAT
BAT


ID 2 motor
2 CABLE-10
CM-510
ID 4 motor
4 CABLE-14
CM-510
ID 4 motor
6 CABLE-10
ID 6 motor
ID 2 motor
8 CABLE-14
ID 8 motor
ID 10 motor
10 CABLE-6
ID 12 motor
ID 8 motor
12 CABLE-18
ID 12 motor
ID 10 motor
14 CABLE-20
ID 14 motor
ID 16 motor
16 CABLE-6
ID 18 motor
ID 14 motor
18 CABLE-20
ID 18 motor

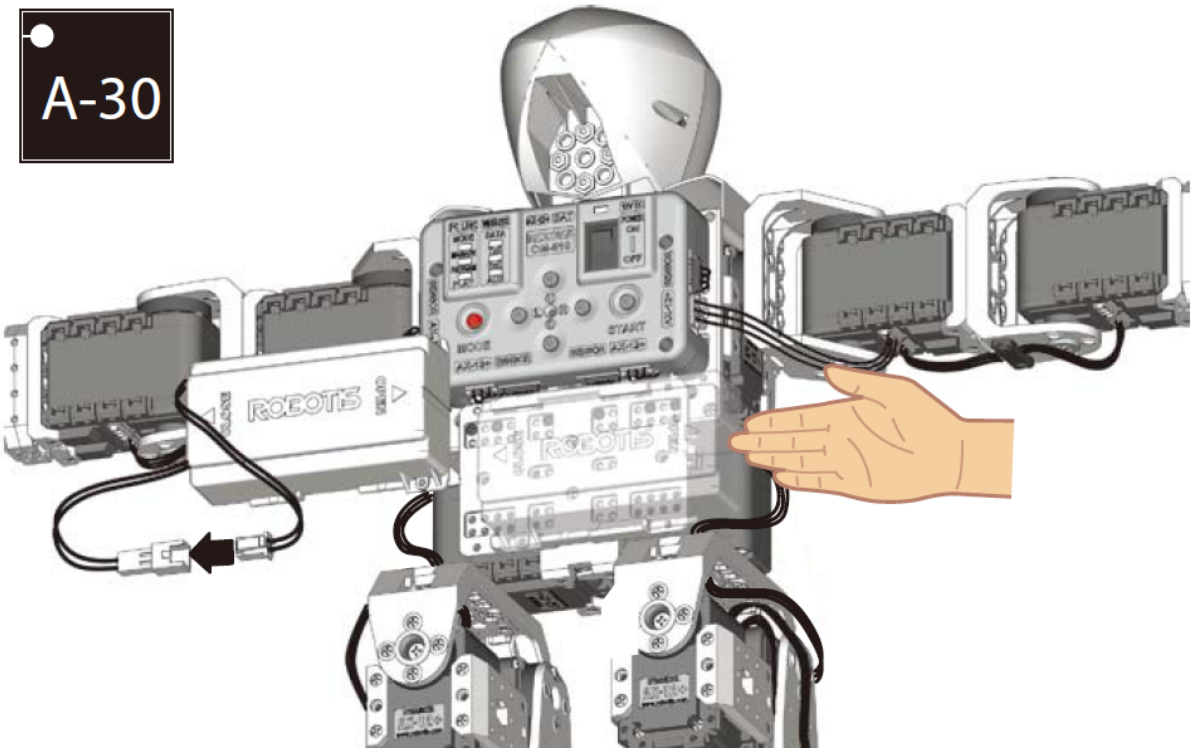
A-27

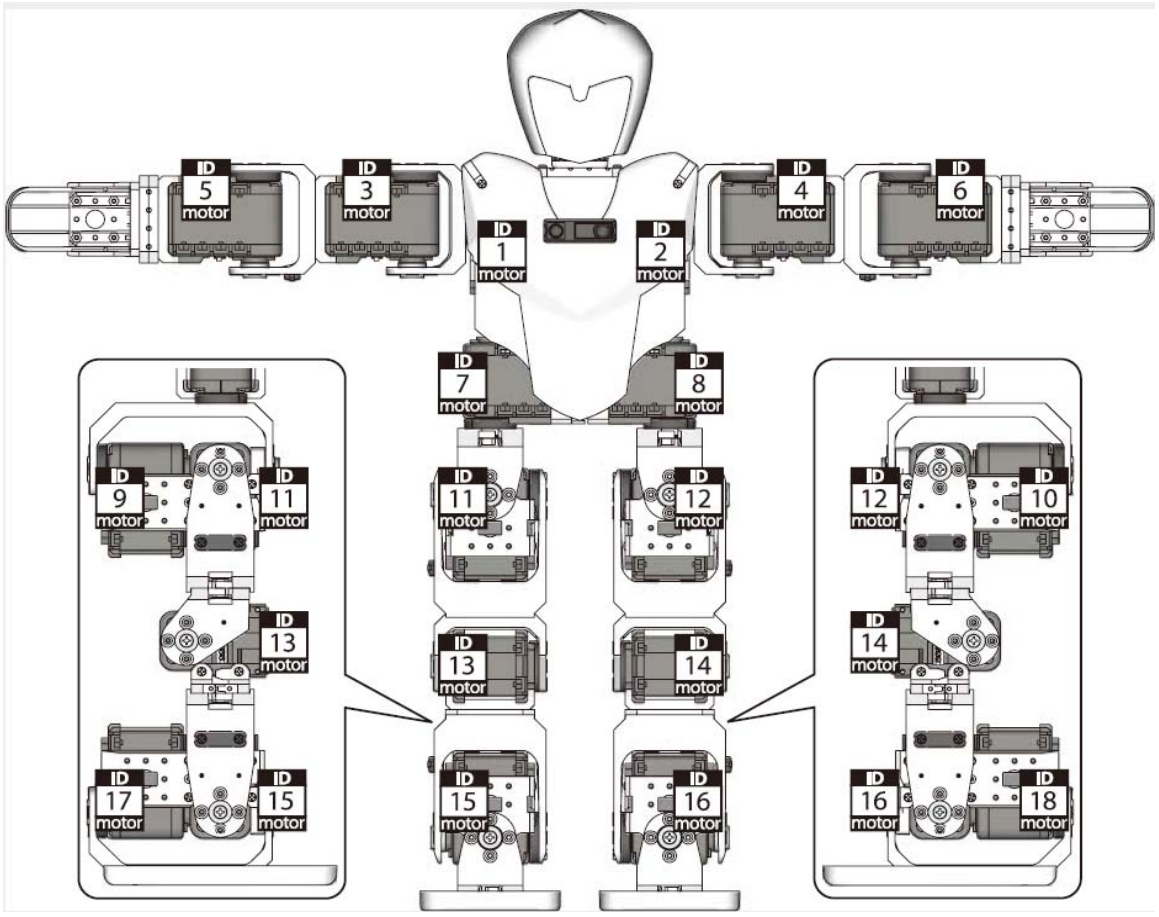

5
3
1
20
2
19
11
4
22
6
13
12
F55 ×8
10
S1 ×4
14
S11 ×12
17
ID 14 motor
N1 ×2
18
16
×2
N1
ID 13 motor


A-28
S12 ×2
×4
S1
×2
S14
F57
×4
N1
IR RECEIVER
RIVET ×6
PLATE 3×7
SINGLE BRACKET ×2
F56
21


A-29
N1 ×2
F58
DMS
S15 ×2
21
S3
×2

A-30


ROBOTIS
ROBOTIS


ID 5 motor
ID 3 motor
ID 1 motor
ID 2 motor
ID 4 motor
ID 6 motor
ID 7 motor
ID 8 motor
ID 9 motor
ID 11 motor
ID 11 motor
ID 12 motor
ID 12 motor
ID 10 motor
ID 13 motor
ID 13 motor
ID 14 motor
ID 14 motor
ID 17 motor
ID 15 motor
ID 15 motor
ID 16 motor
ID 16 motor
ID 18 motor

4.2. Robot Dinosaurus

PRAKTICKÁ ÚLOHA

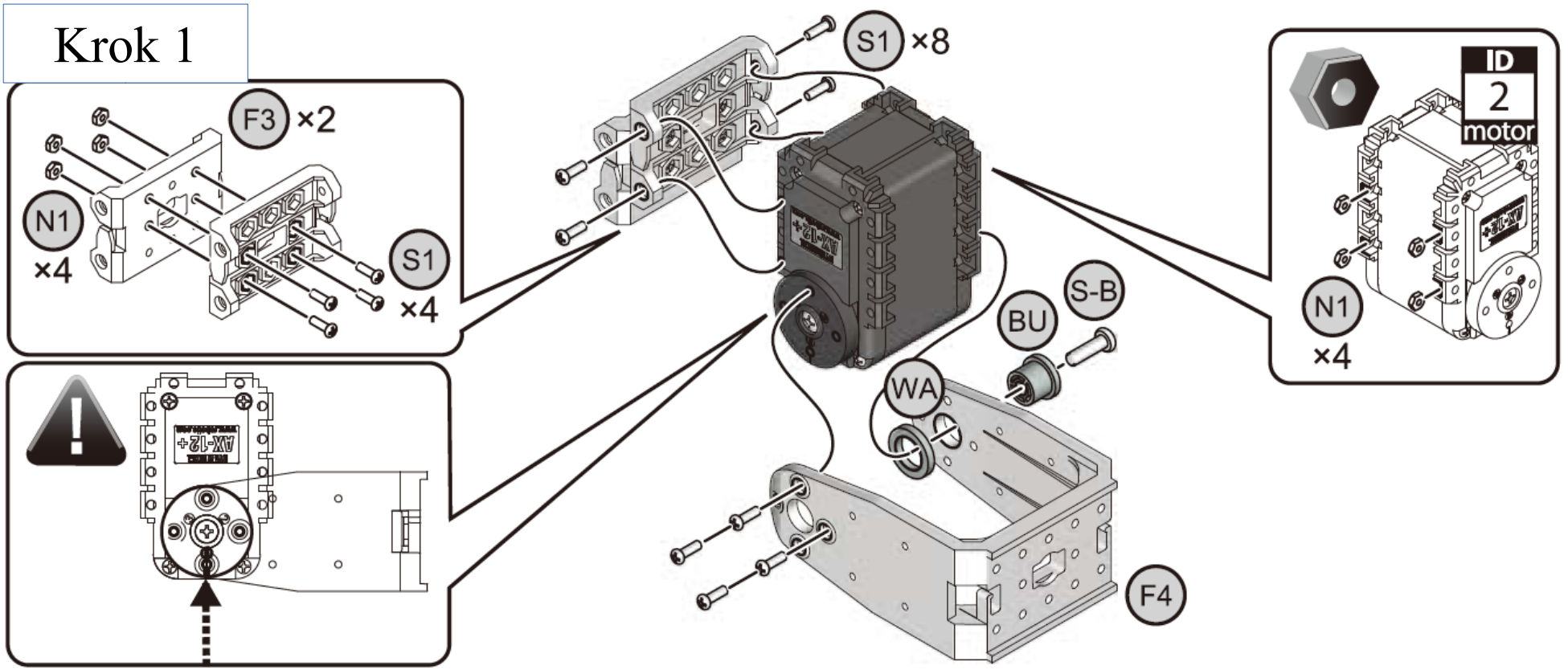
Pomocí této kapitoly lze sestavit mobilního robota, který má čtyři končetiny a svým tvarem připomíná pravěkého dinosaura. Robot je tvořen 15-ti pohony AX-12. Horní končetiny jsou tvořeny dvěma pohony AX-12 a dolní končetiny jsou tvořeny třemi pohony AX-12. Robot je tvořen ocasem, který může být použitý ke stabilizaci pohybu.

ČAS K SESTAVENÍ: 620 minut

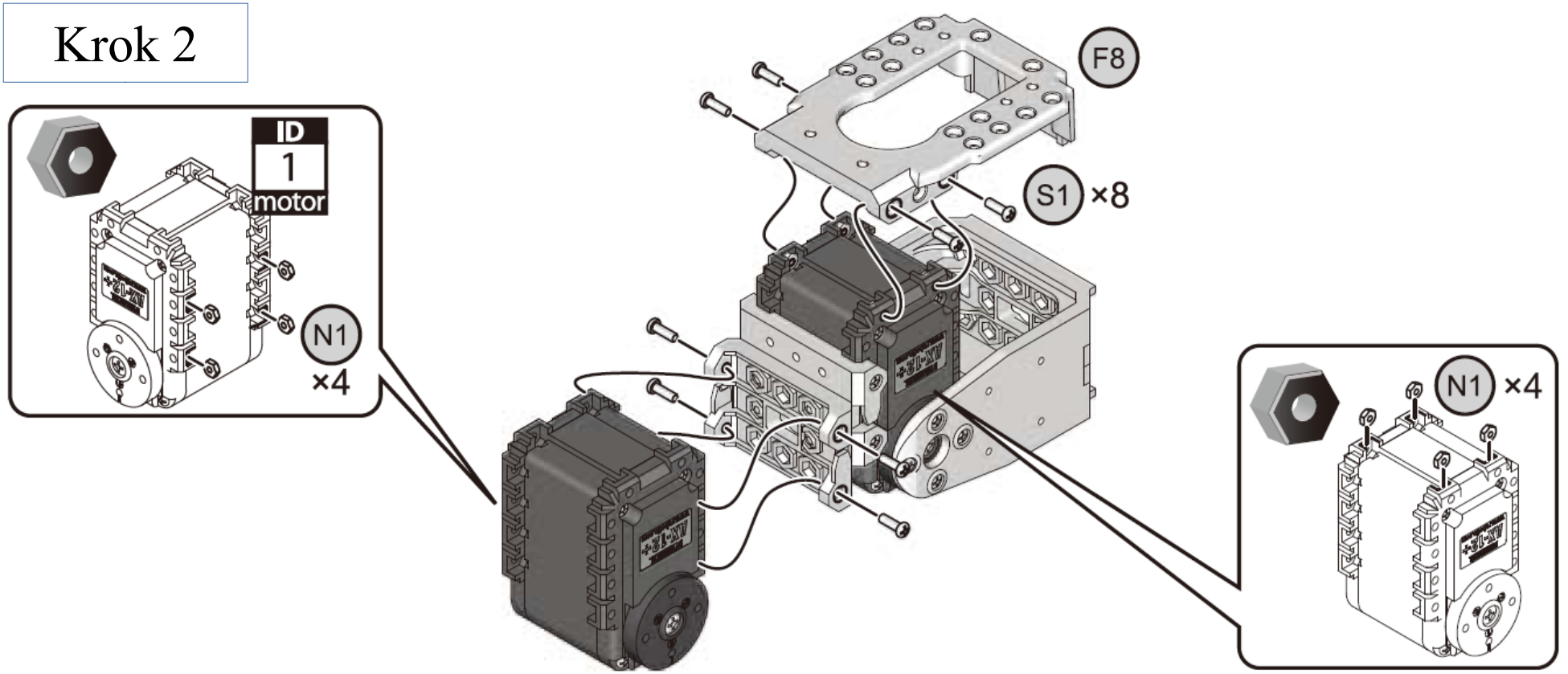
CÍL

Cílem této úlohy je ukázat simulaci základních pohybů pravěkého zvířete. Vzhledem k tomu, že chůze po dvou končetinách je obtížná k realizaci, je tento robot doplněn ocasem, který tak může sloužit ke stabilizaci chůze.

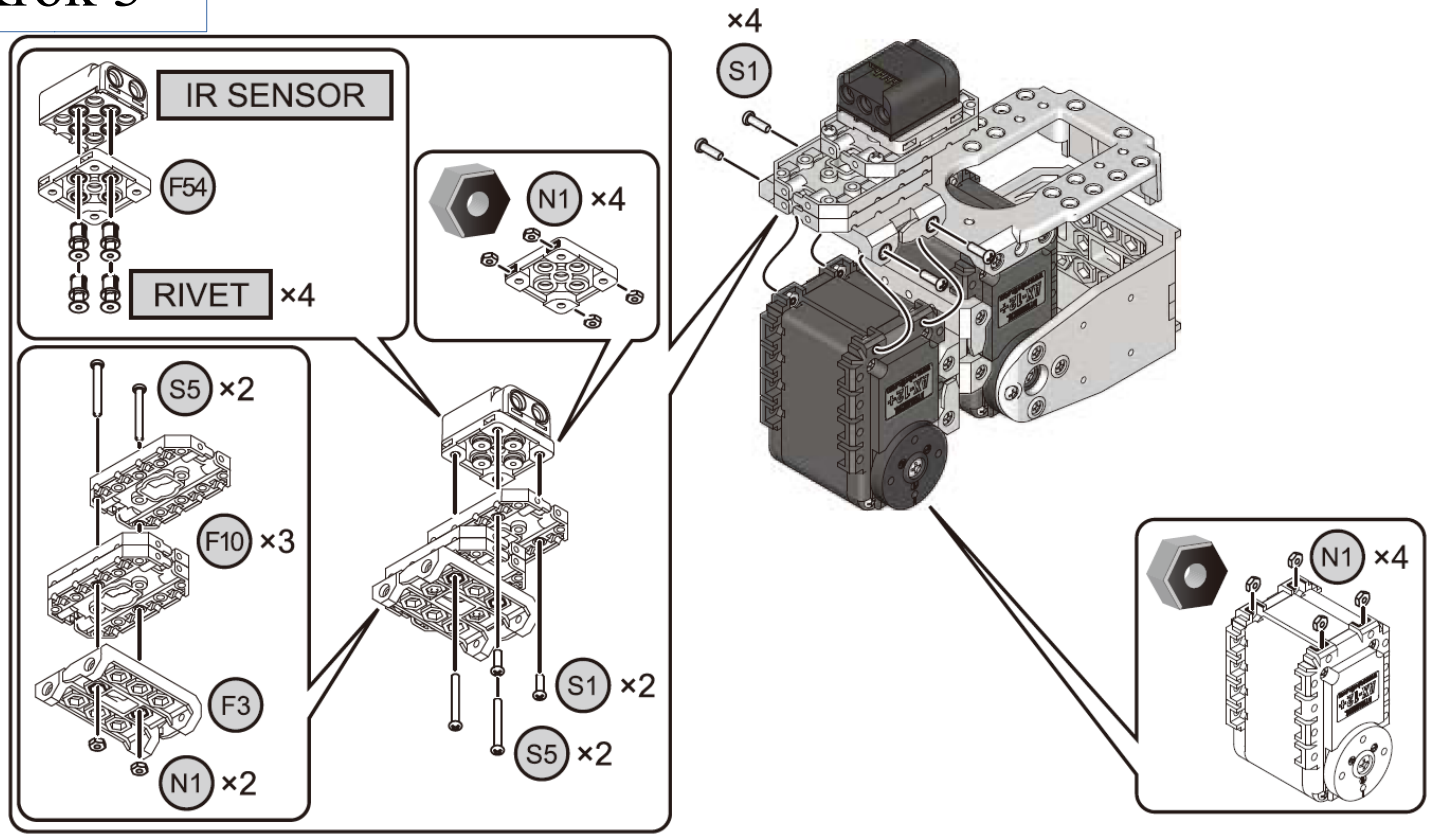
Postup sestavení robota:



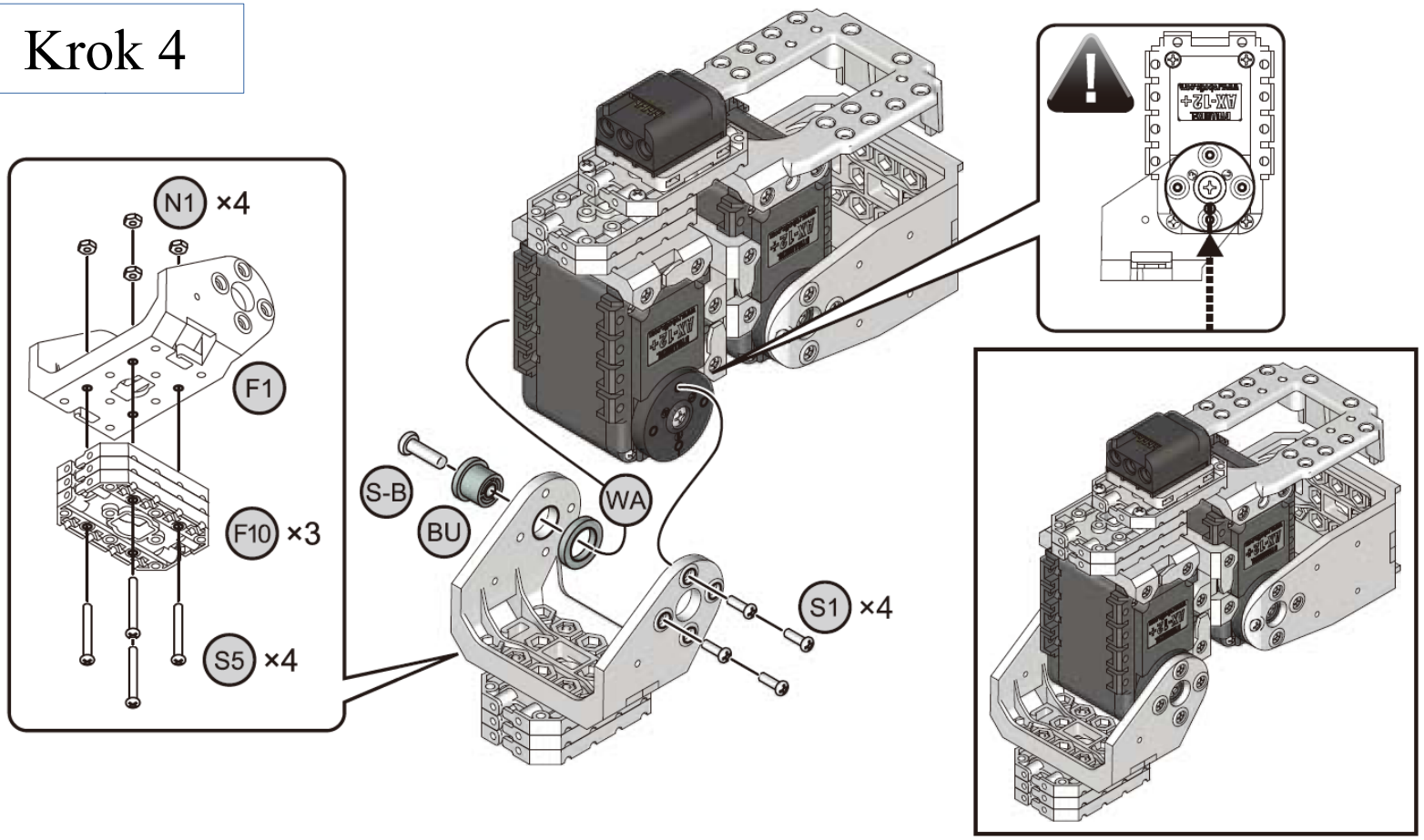
Krok 2



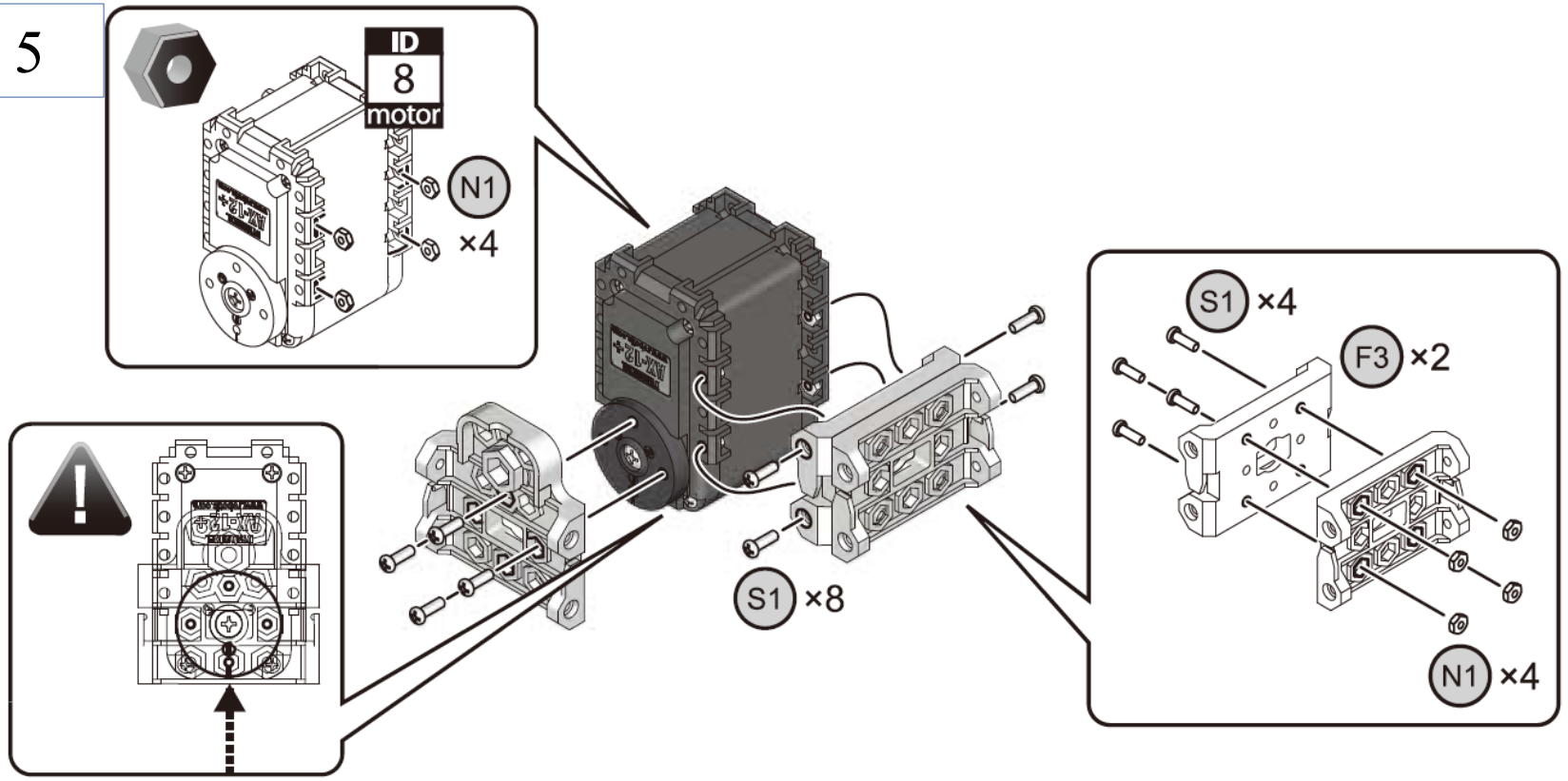
Krok 3



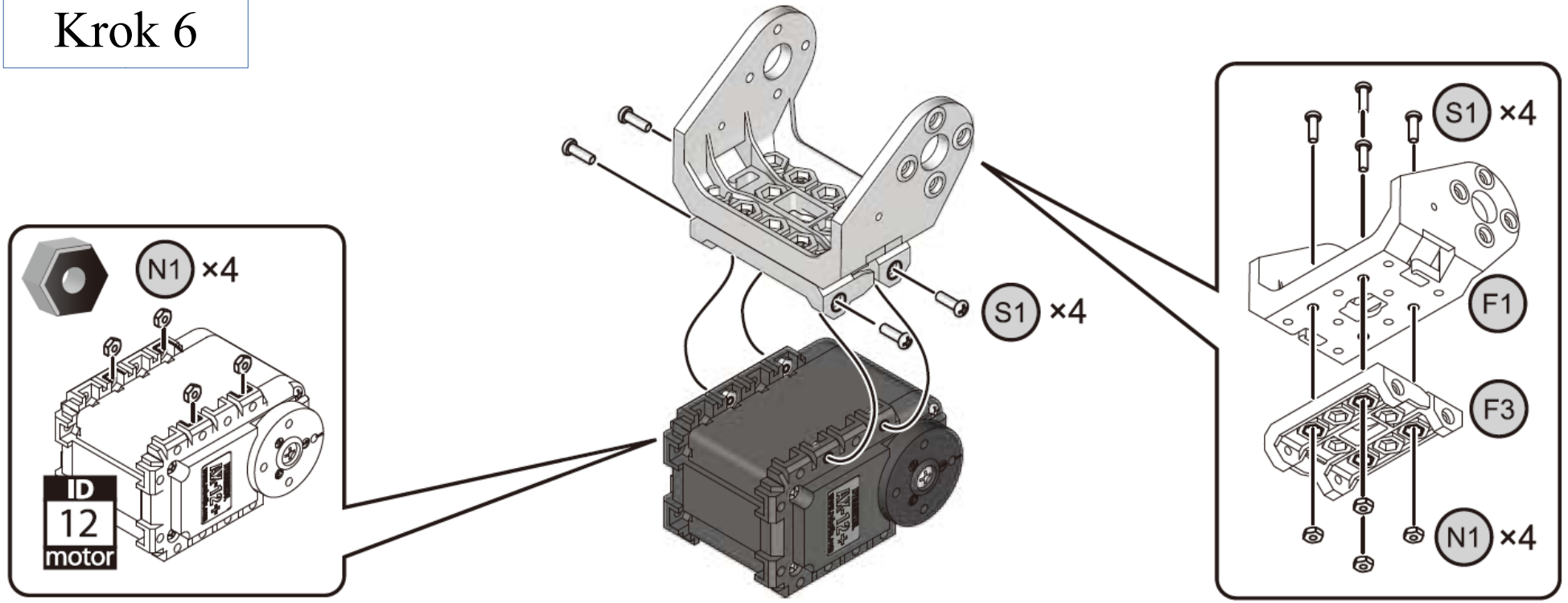
Krok 4



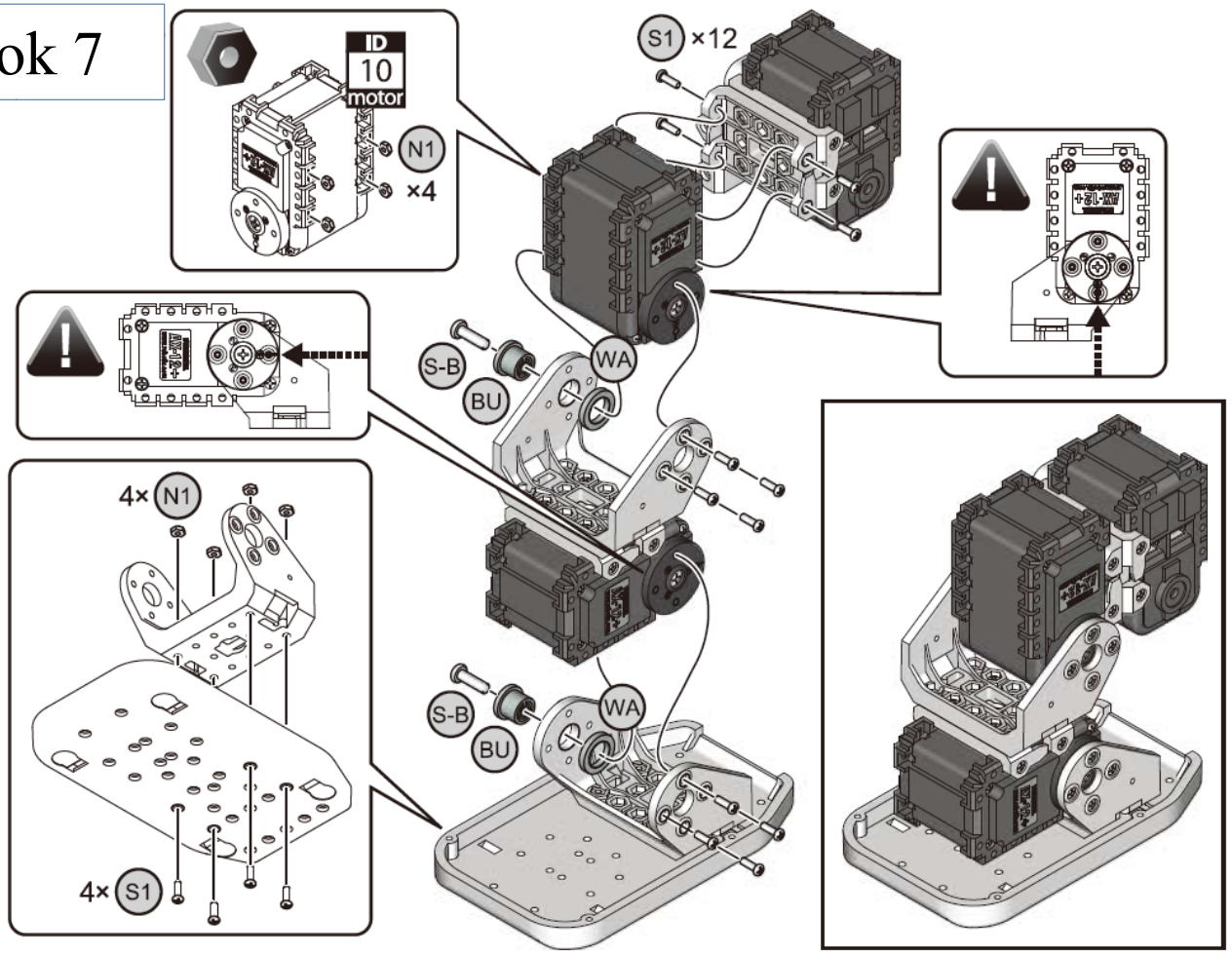
Krok 5



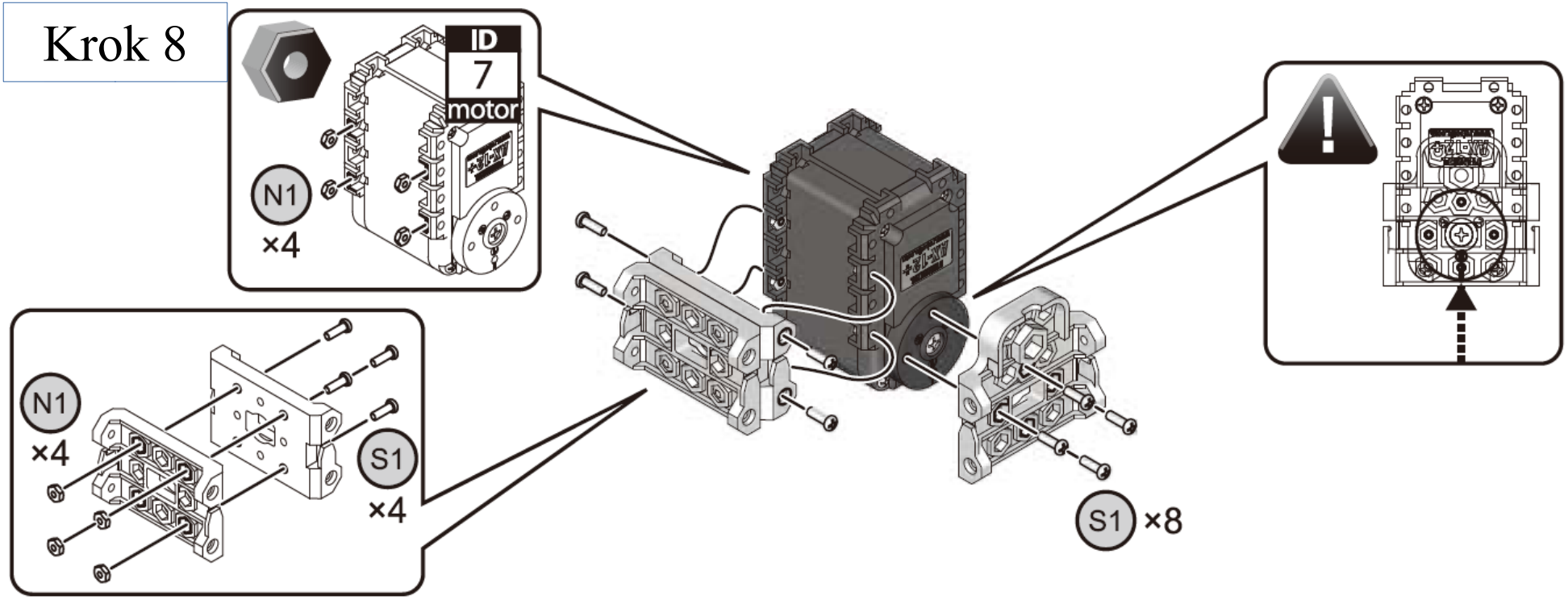
Krok 6



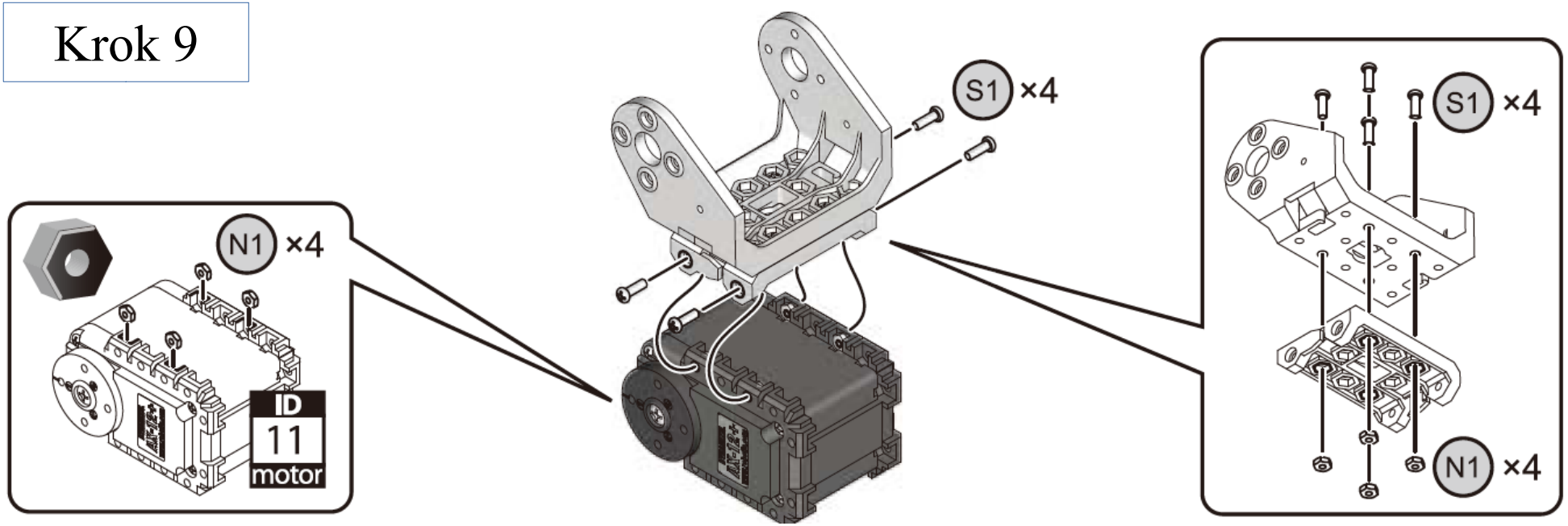
Krok 7



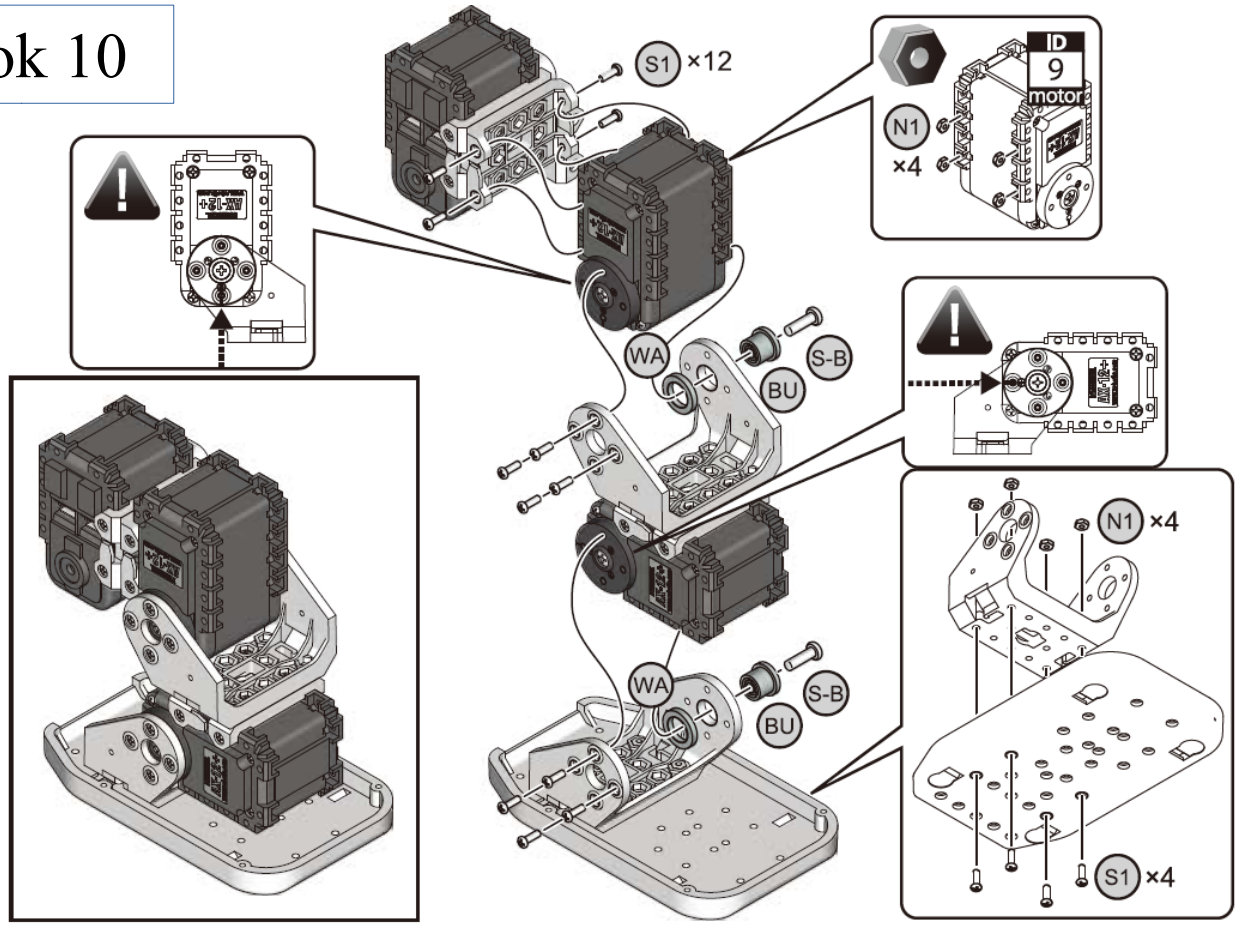
Krok 8



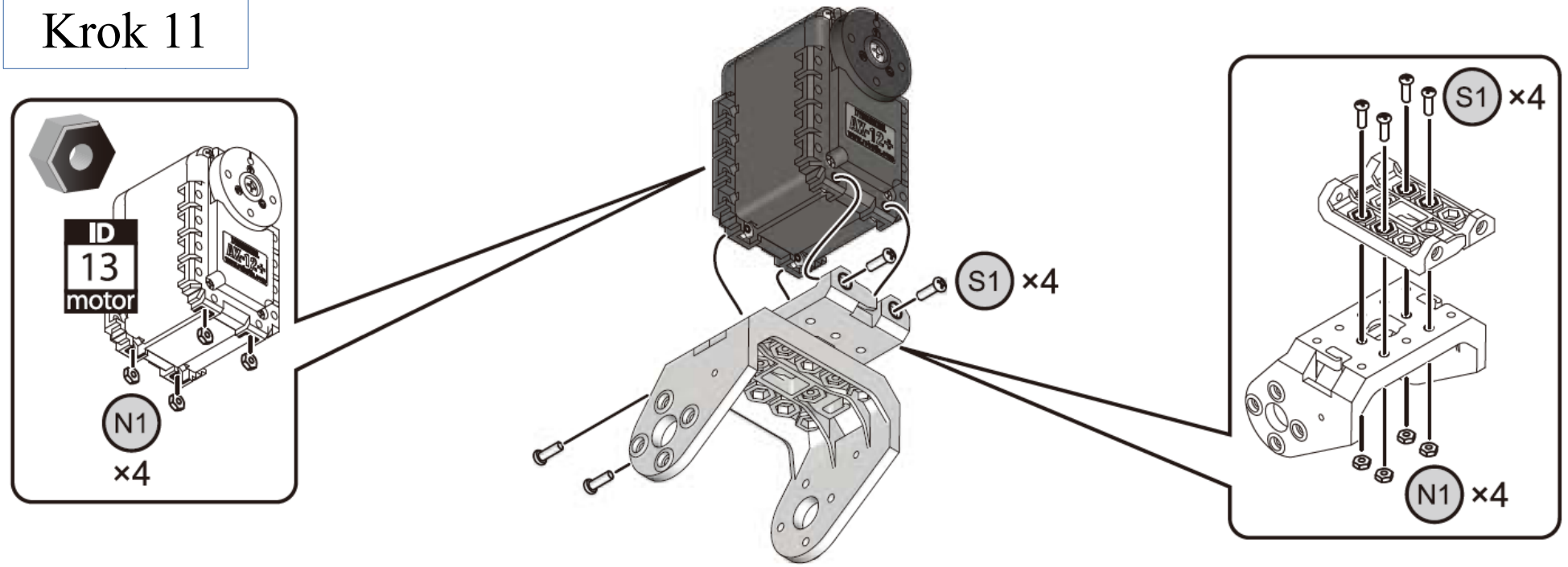
Krok 9



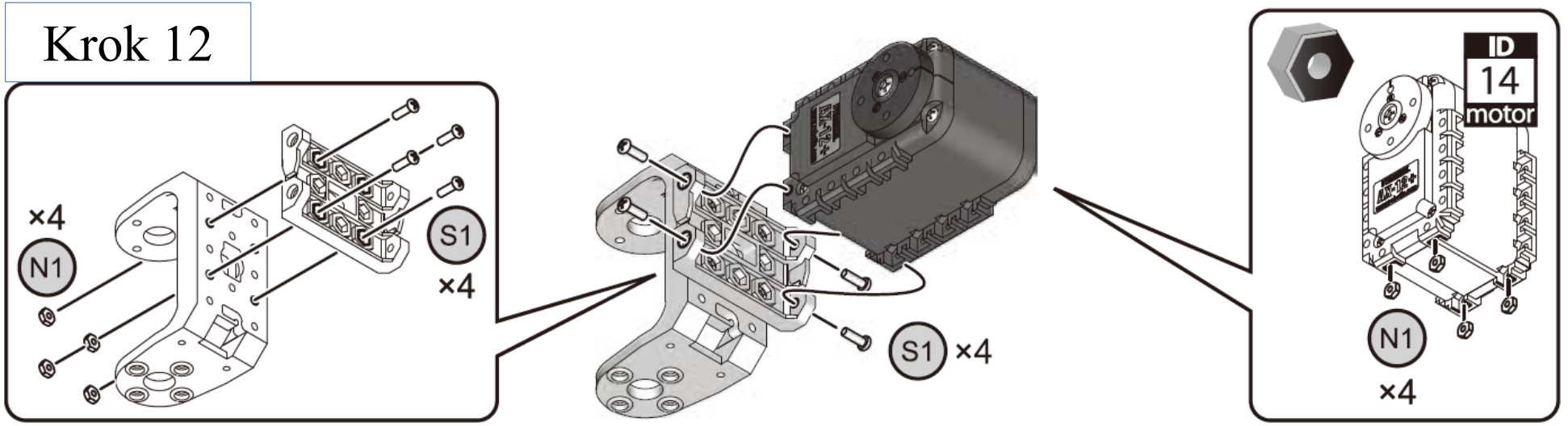
Krok 10



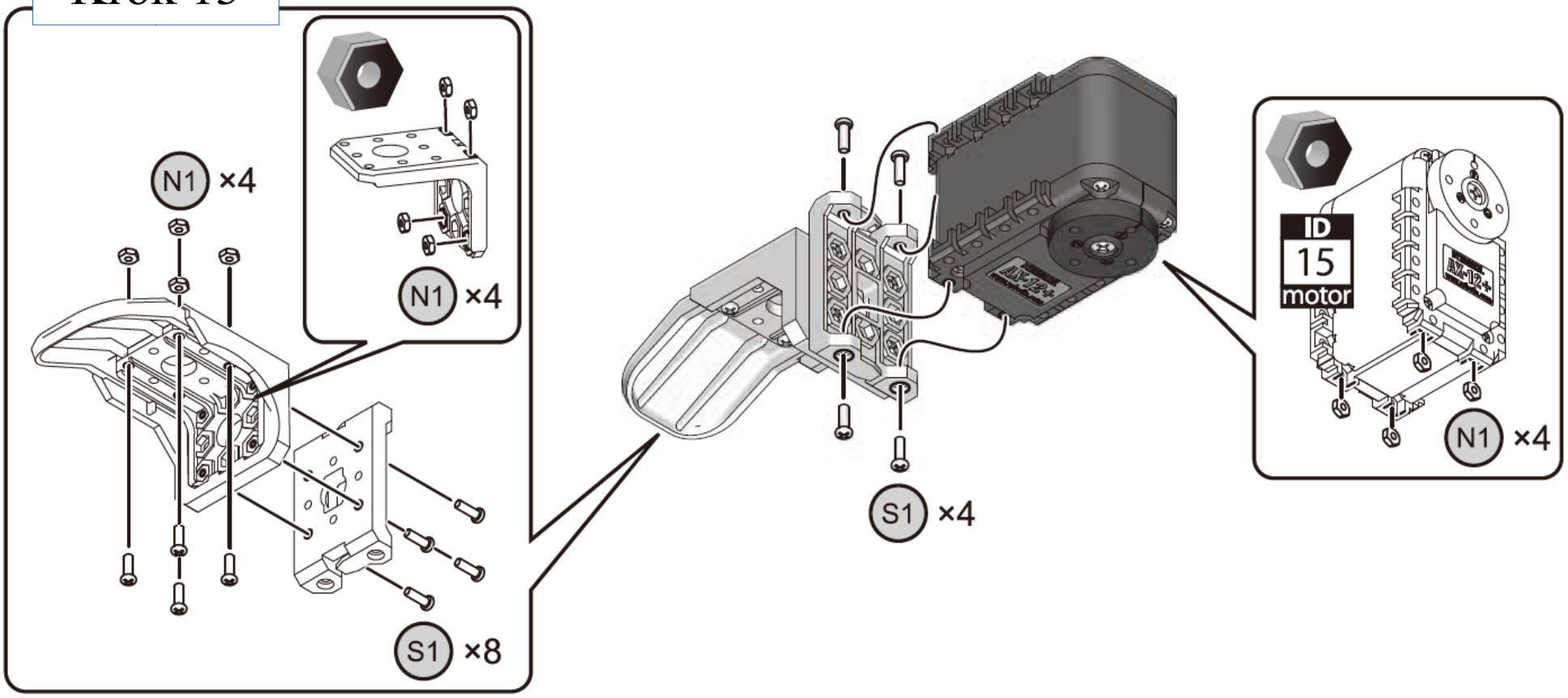
Krok 11



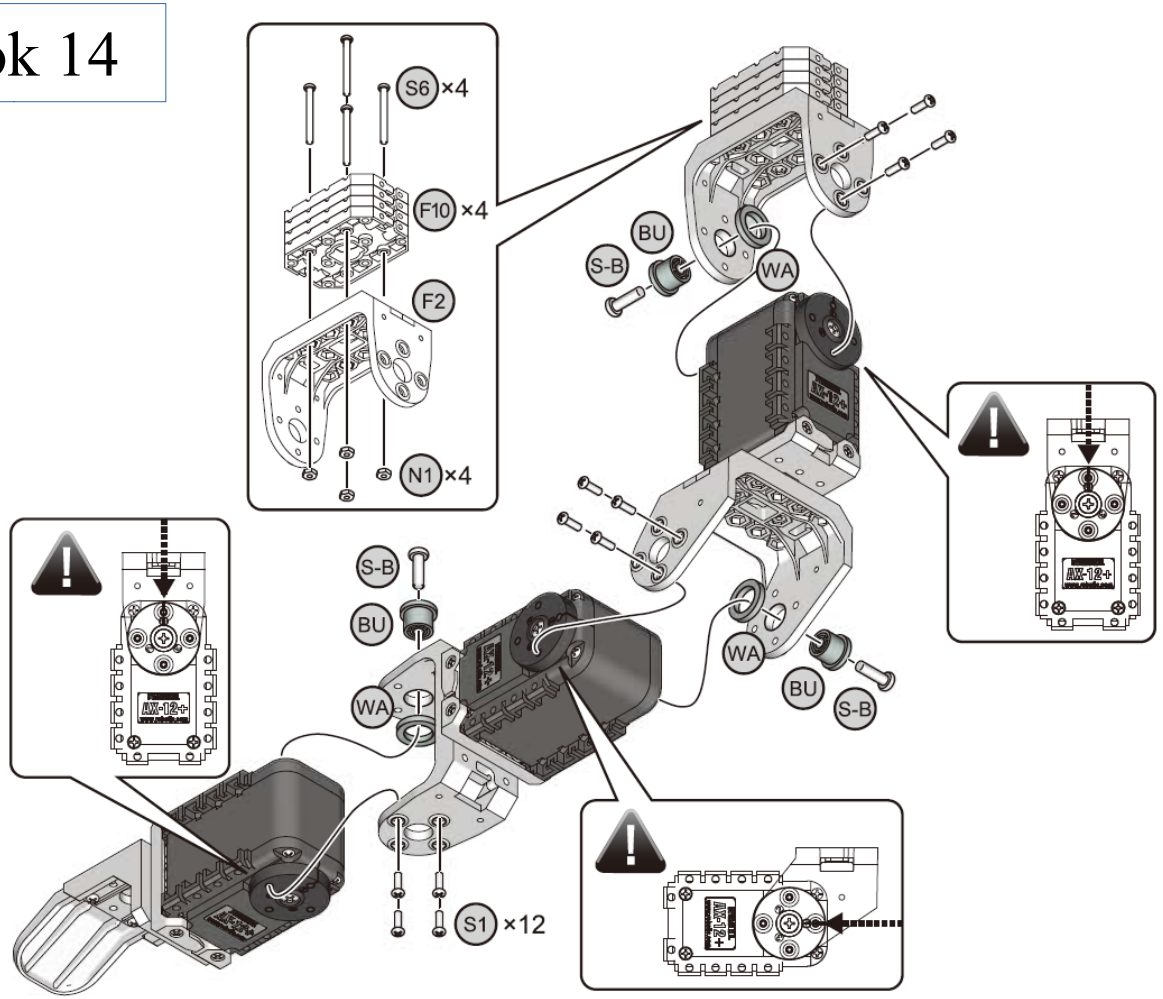
Krok 12



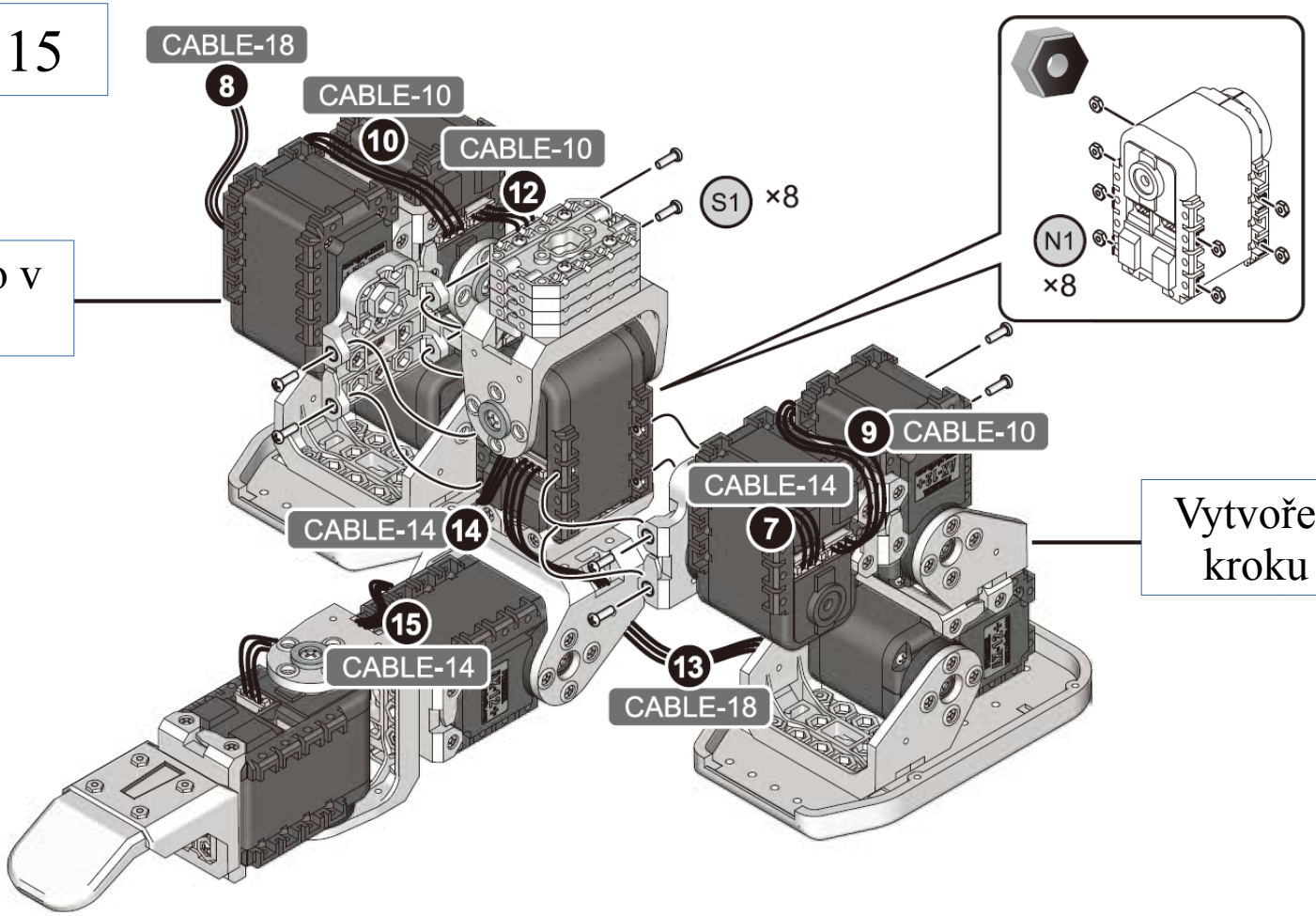
Krok 13



Krok 14



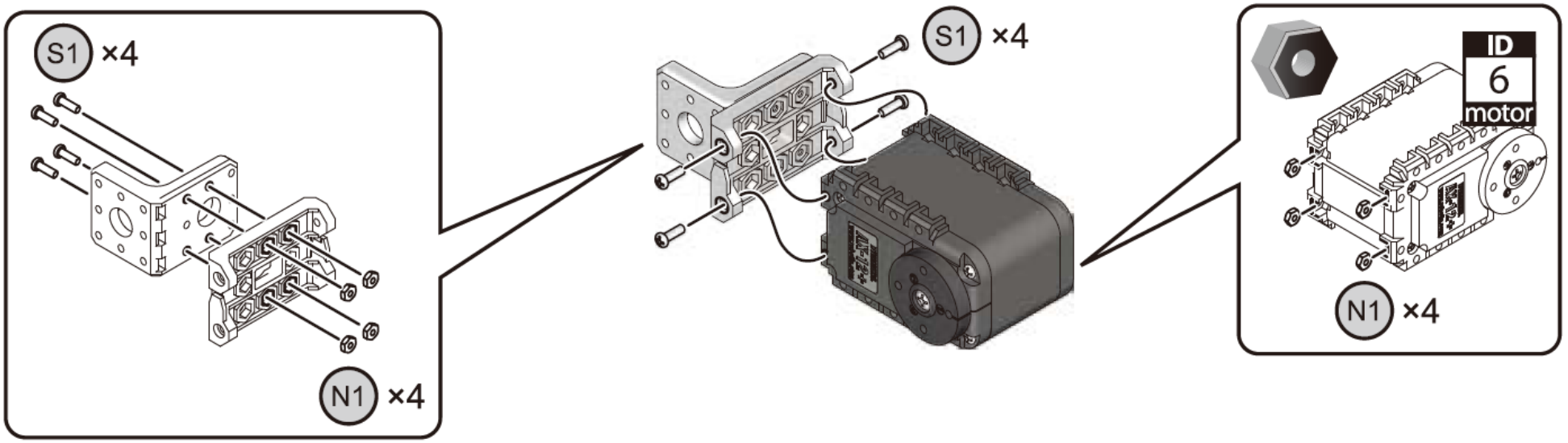
Krok 15



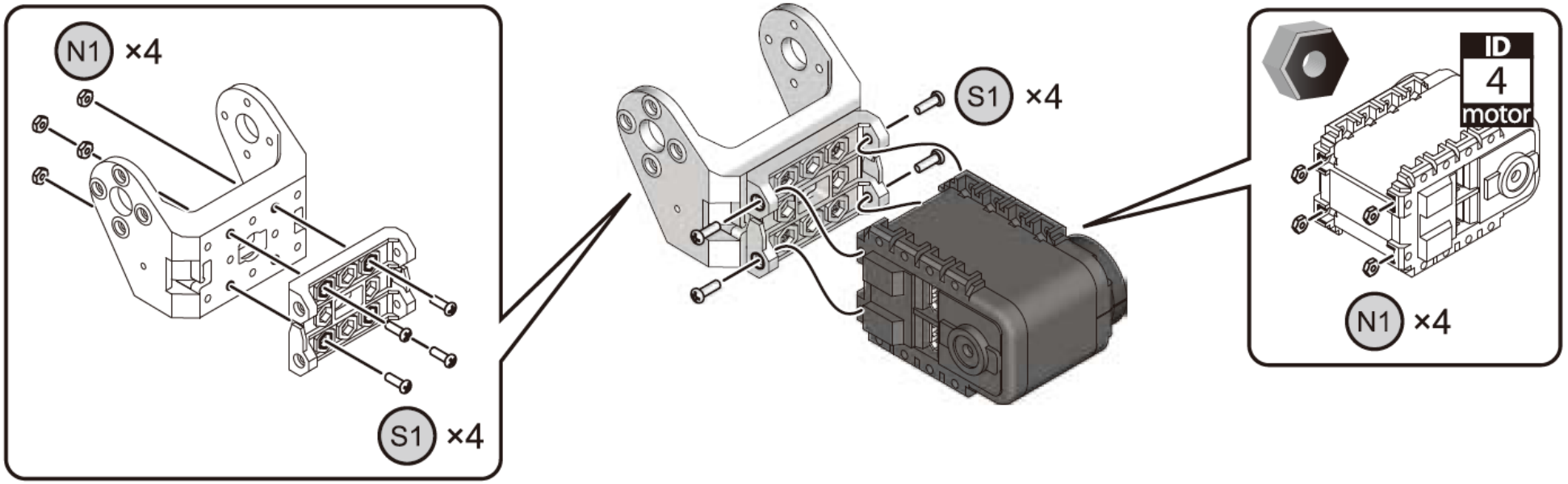
Vytvořeno v kroku 7

Vytvořeno v kroku 10

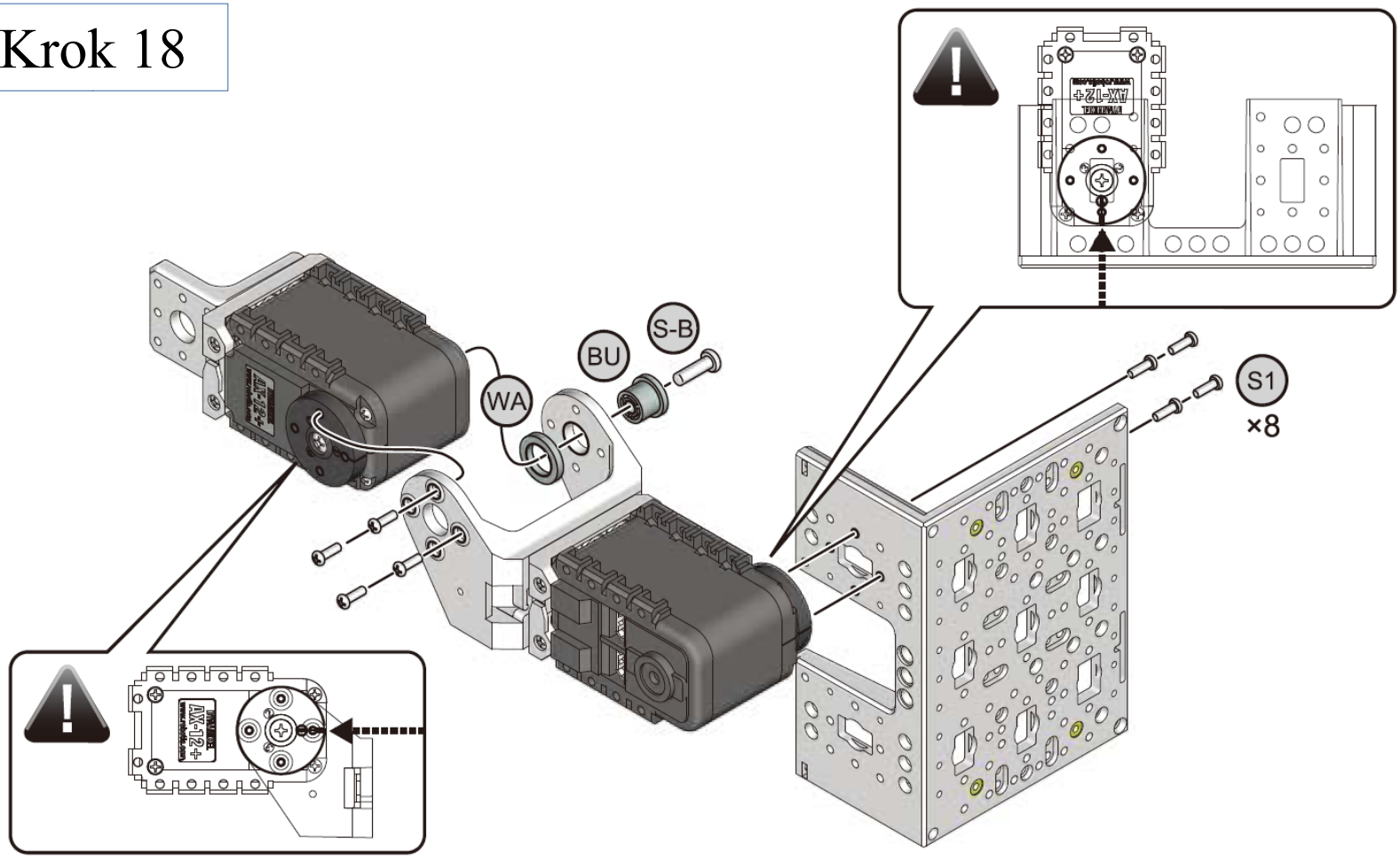
Krok 16



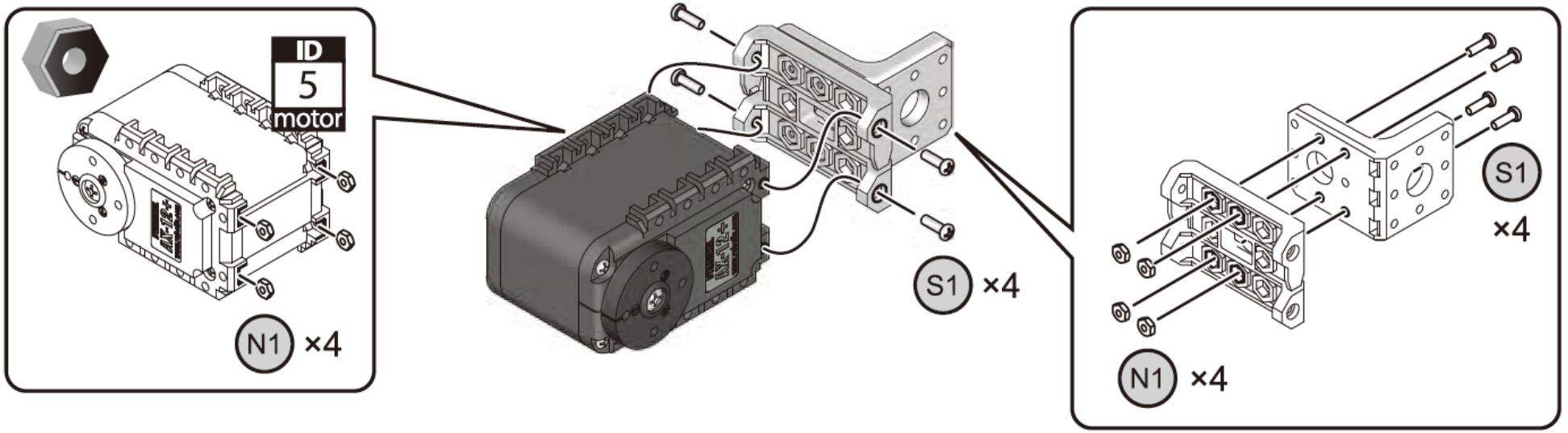
Krok 17



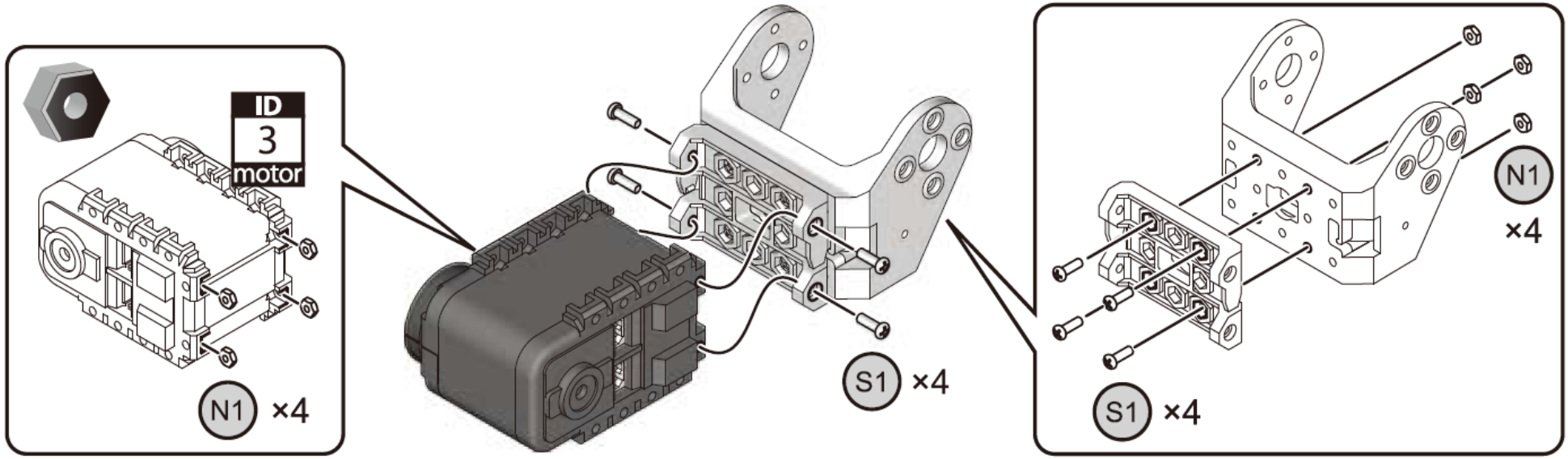
Krok 18



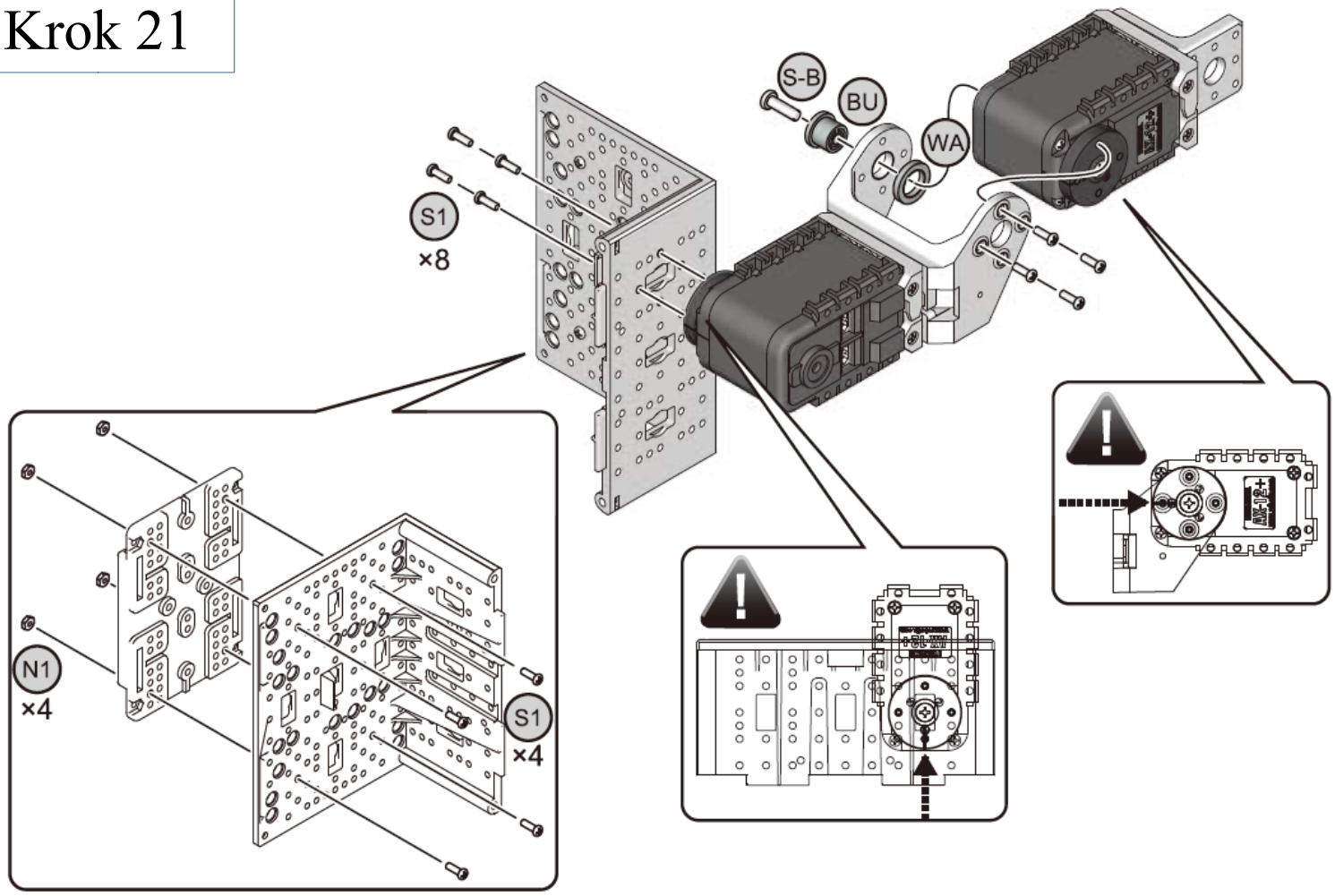
Krok 19



Krok 20

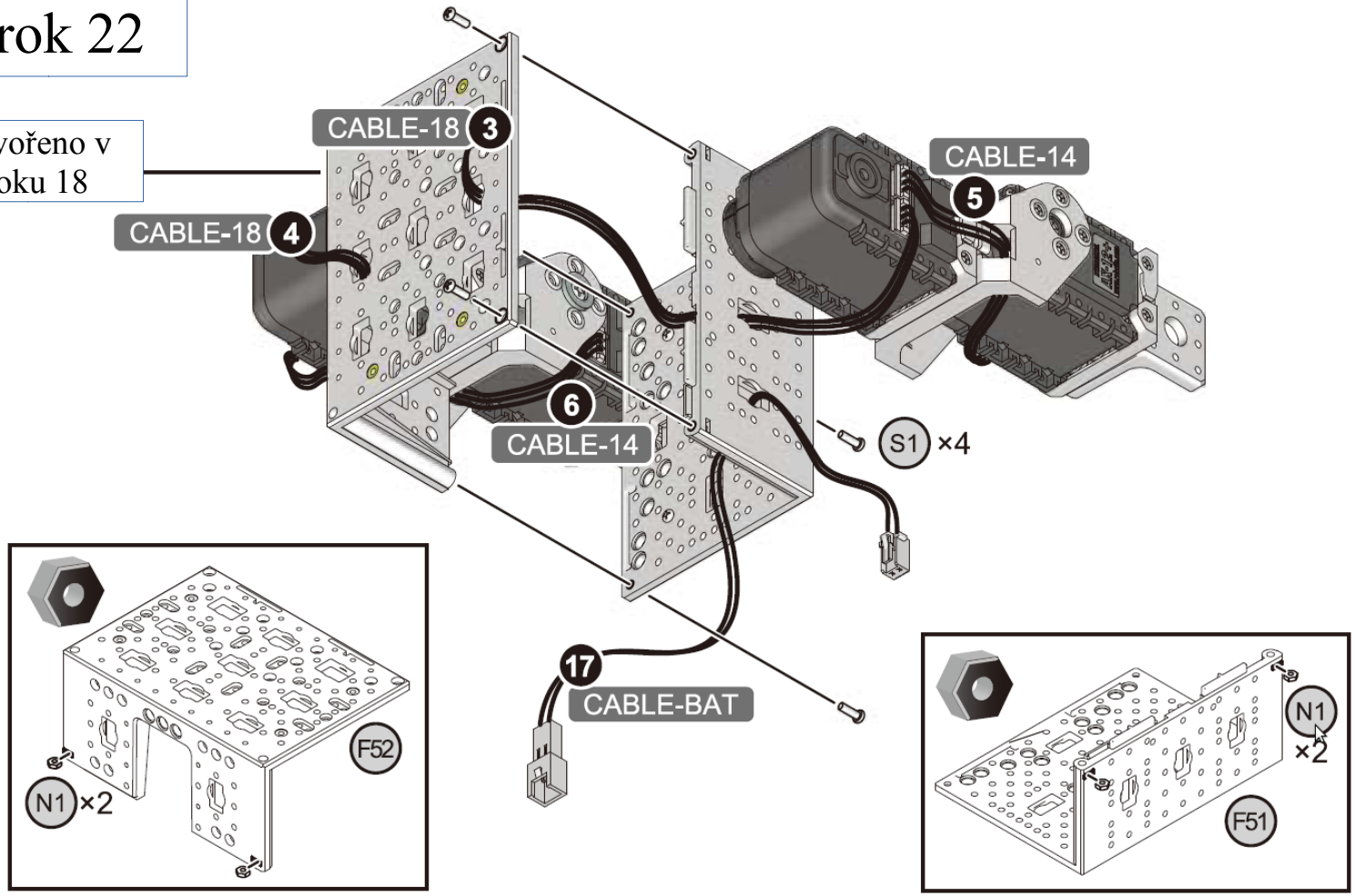


Krok 21

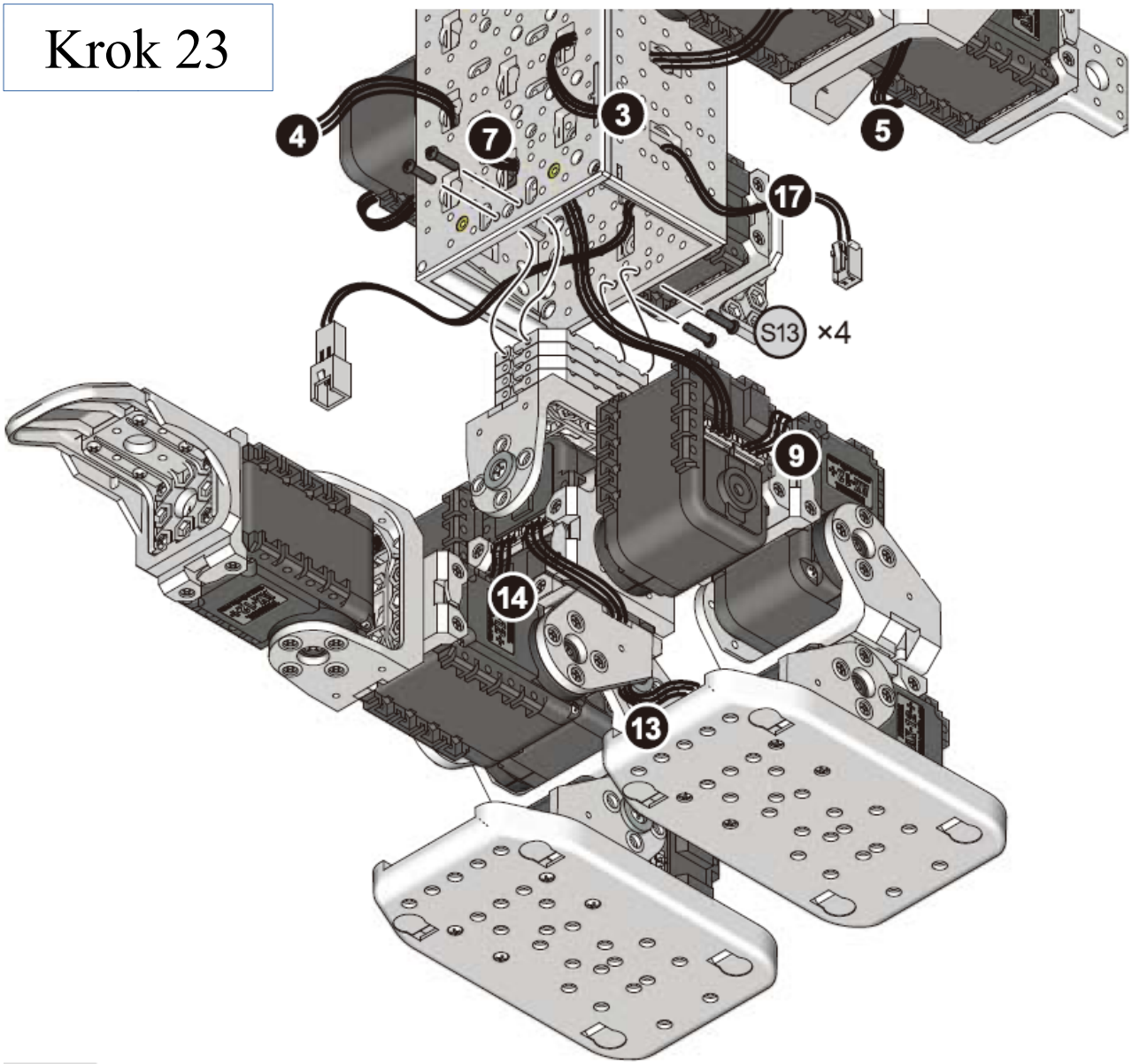


Krok 22

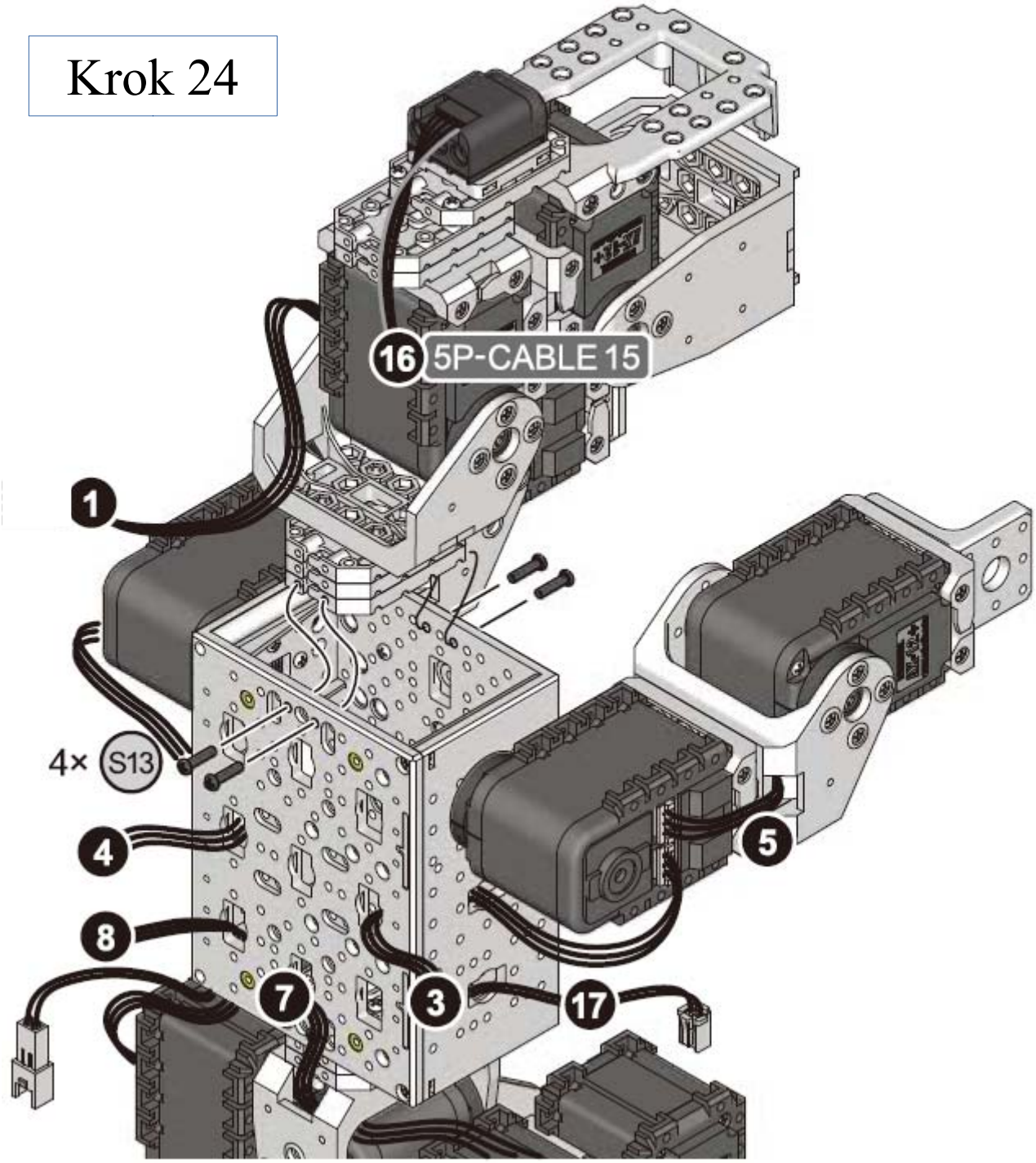
Vytvořeno v kroku 18



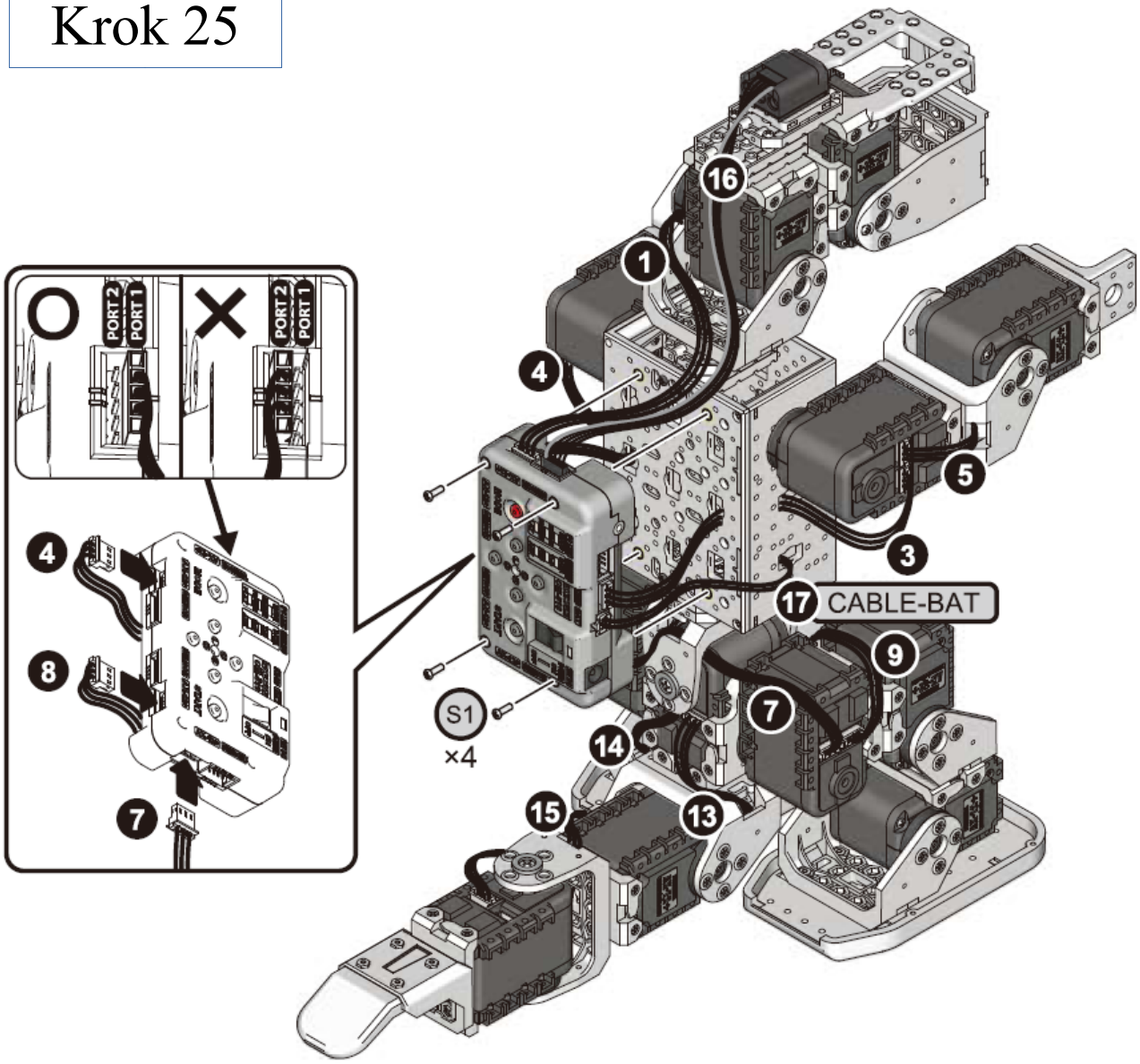
Krok 23

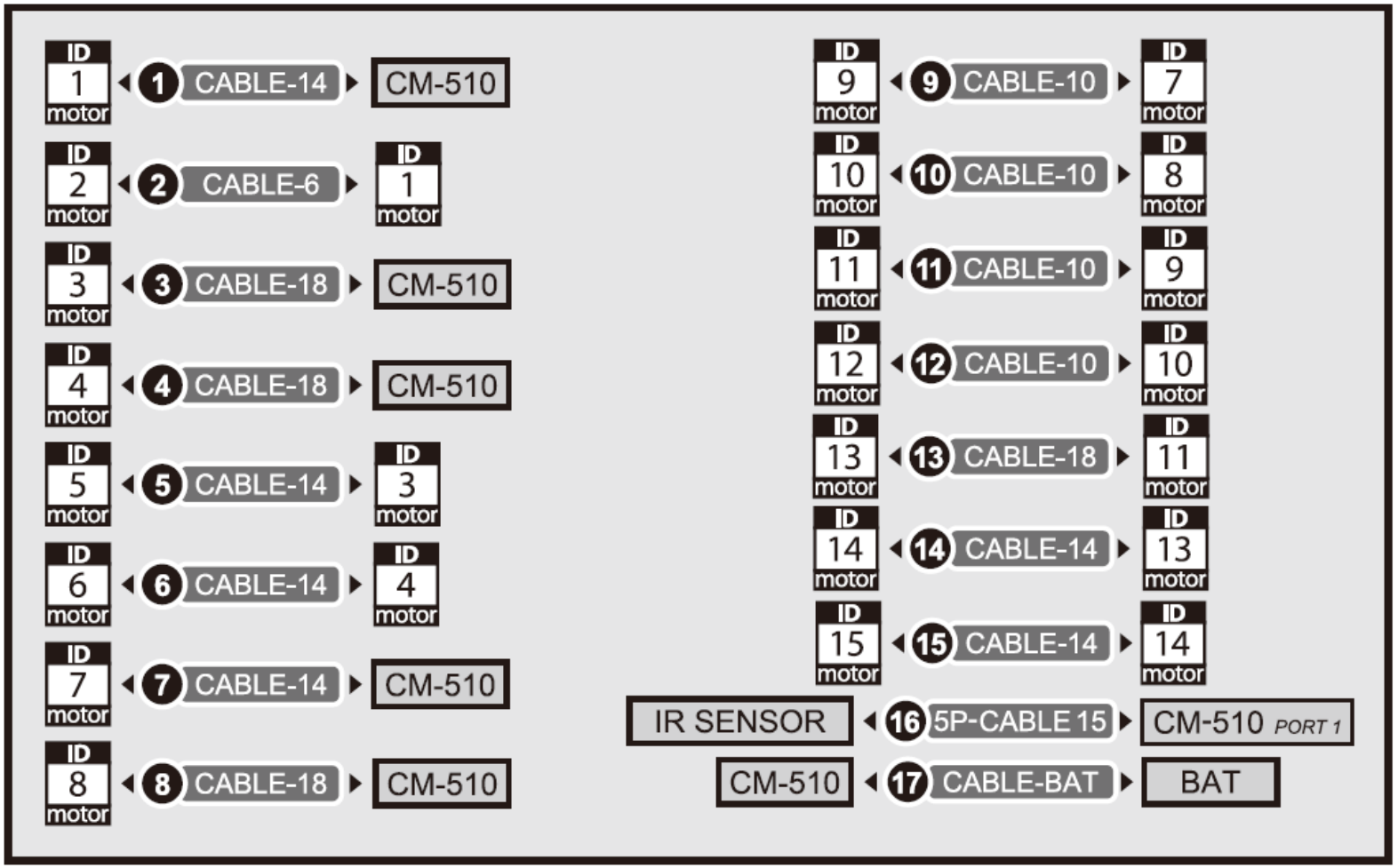


Krok 24

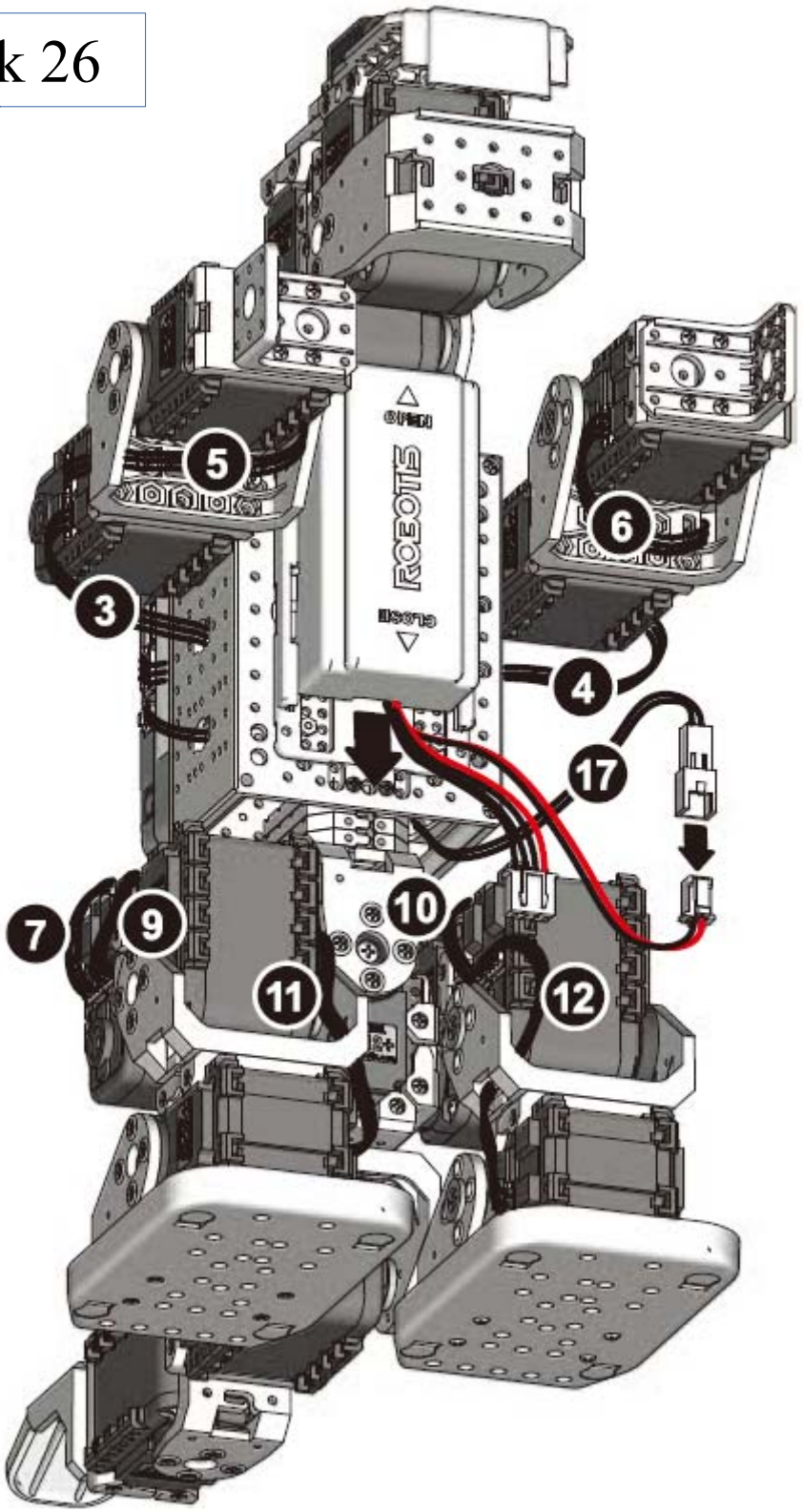


Krok 25

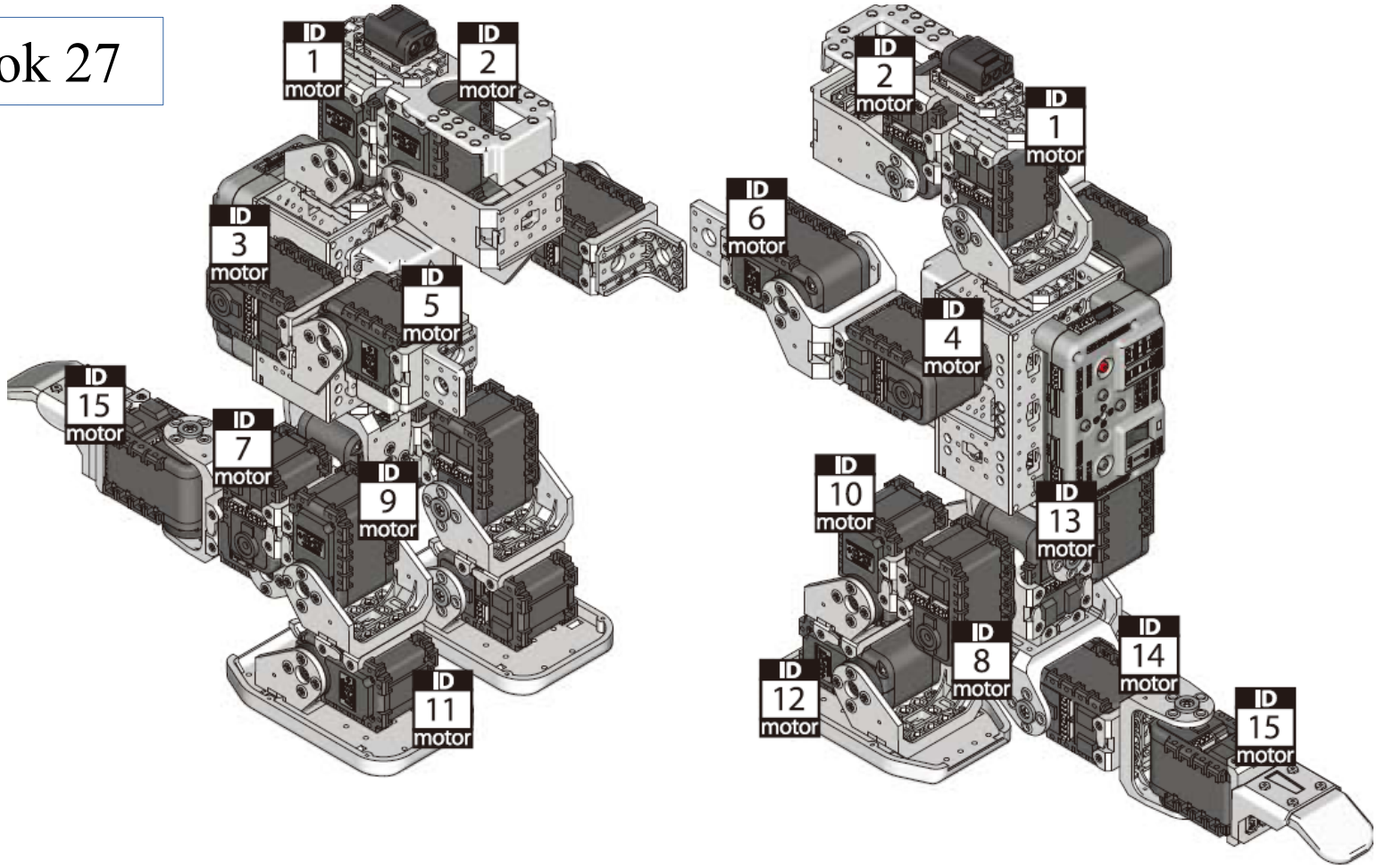

PORT 2
PORT 1
PORT 2
PORT 1
16
1
4
5
3
17 CABLE-BAT
9
7
S1
×4
14
15
13
4
8
7


ID 1 motor
1 CABLE-14
CM-510
ID 2 motor
2 CABLE-6
ID 1 motor
ID 3 motor
3 CABLE-18
CM-510
ID 4 motor
4 CABLE-18
CM-510
ID 5 motor
5 CABLE-14
ID 3 motor
ID 6 motor
6 CABLE-14
ID 4 motor
ID 7 motor
7 CABLE-14
CM-510
ID 8 motor
8 CABLE-18
CM-510
ID 9 motor
9 CABLE-10
ID 7 motor
ID 10 motor
10 CABLE-10
ID 8 motor
ID 11 motor
11 CABLE-10
ID 9 motor
ID 12 motor
12 CABLE-10
ID 10 motor
ID 13 motor
13 CABLE-18
ID 11 motor
ID 14 motor
14 CABLE-14
ID 13 motor
ID 15 motor
15 CABLE-14
ID 14 motor
IR SENSOR
16 5P-CABLE 15
CM-510 PORT 1
CM-510
17 CABLE-BAT
BAT

Krok 26



Krok 27


ID 1 motor
ID 2 motor
ID 3 motor
ID 5 motor
ID 15 motor
ID 7 motor
ID 9 motor
ID 11 motor
ID 2 motor
ID 1 motor
ID 6 motor
ID 4 motor
ID 10 motor
ID 13 motor
ID 8 motor
ID 14 motor
ID 12 motor
ID 15 motor

4.3. Robot Pes


PRAKTICKÁ ÚLOHA

Pomocí této kapitoly lze sestavit mobilního robota, který má čtyři končetiny a svým tvarem připomíná malého psa nebo štěně. Robot je tvořen 15-ti pohony AX-12. Každá končetina je tvořena třemi pohony AX-12, díky čemuž je možné použít tohoto robota k nácviku chůze. Vzhledem ke čtyřem končetinám je práce s tímto robotem o něco snadnější než u robota Humanoid.

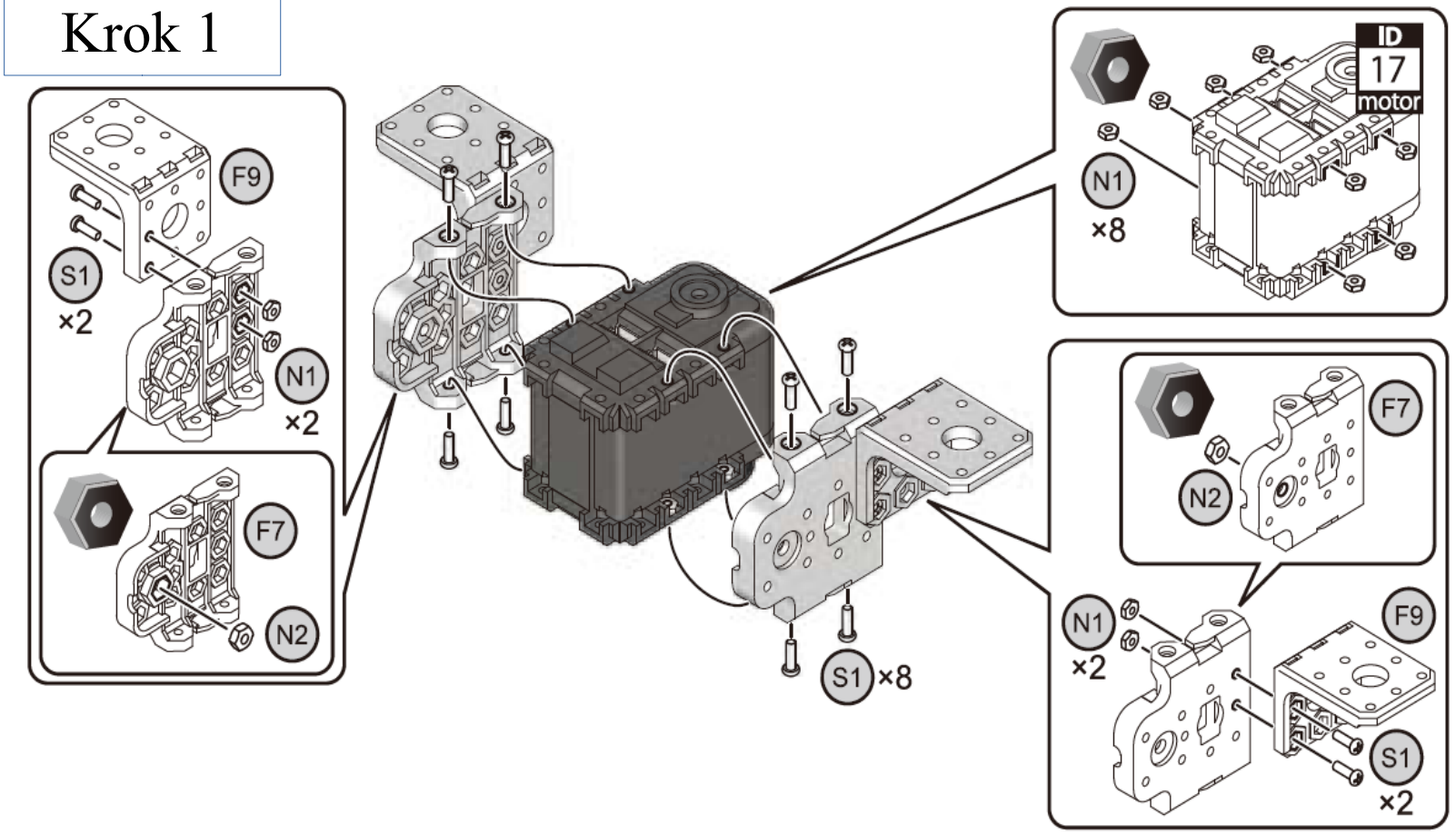

ČAS K SESTAVENÍ: 620 minut


CÍL

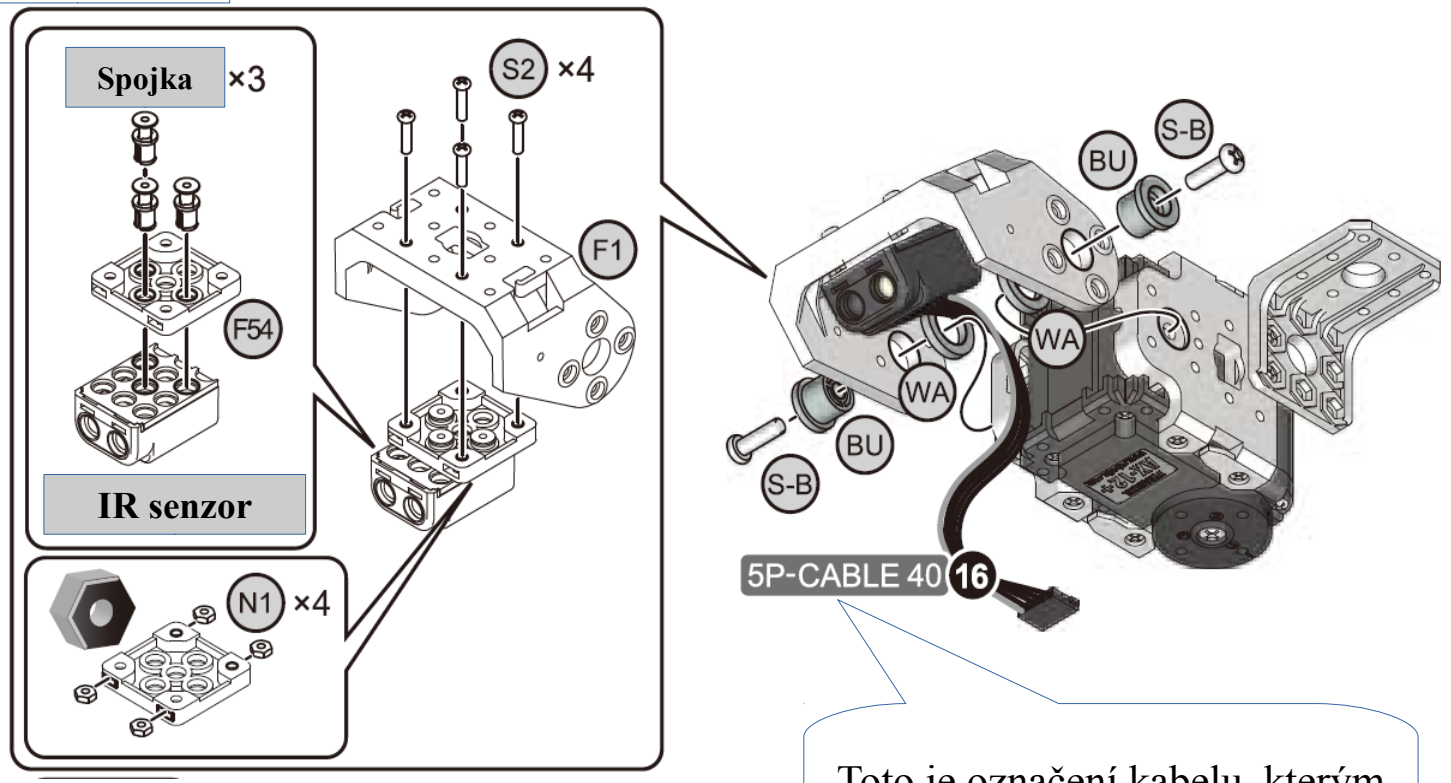
Cílem této úlohy je ukázat simulaci základních pohybů zvířete. Dalším cílem této úlohy je ukázka práce s těžištěm, jehož určení je nezbytné pro správné vytvoření algoritmů, simulujících chůzi čtyř nohého zvířete.

Postup sestavení robota:

Krok 1

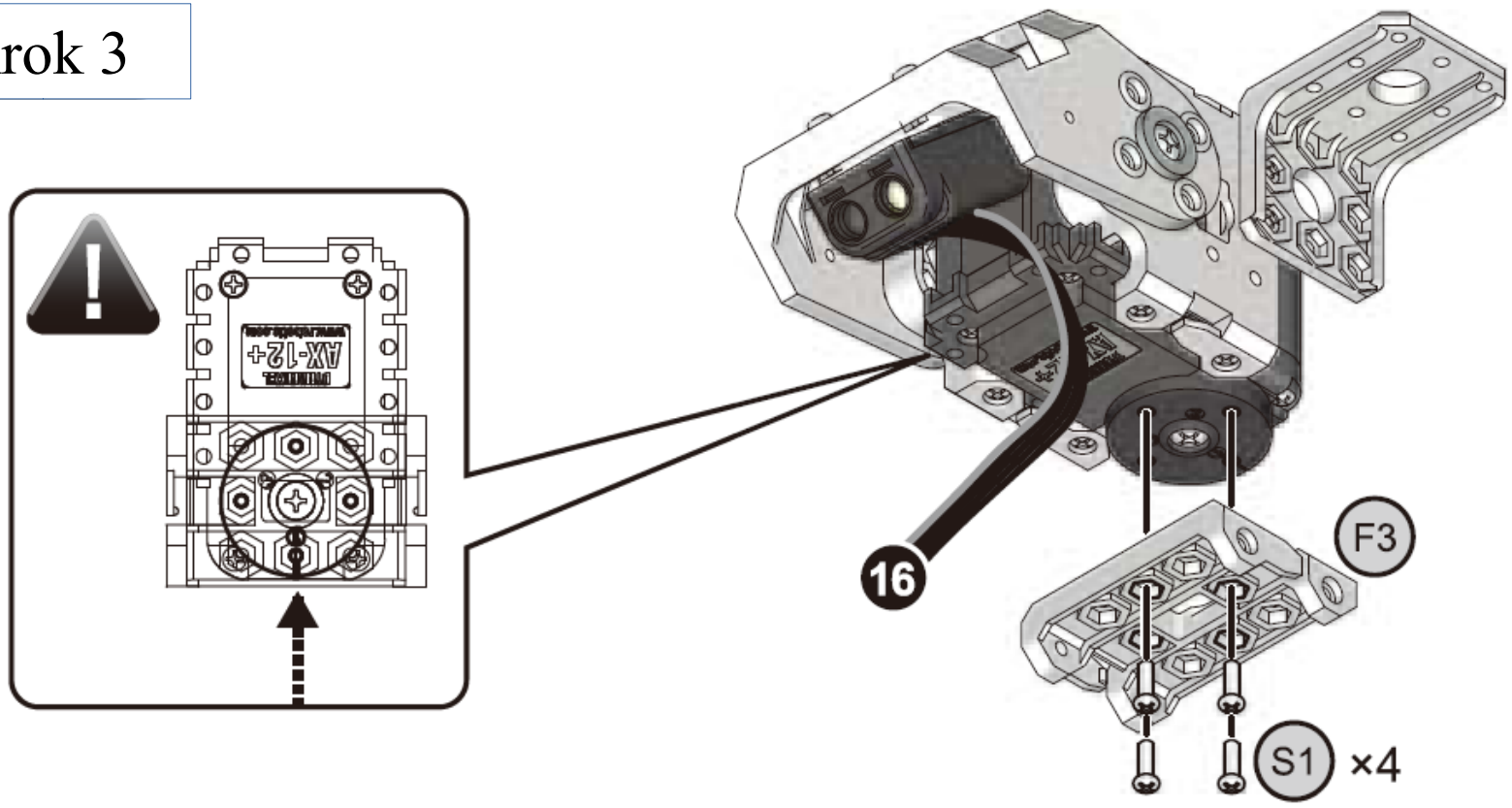


Krok 2

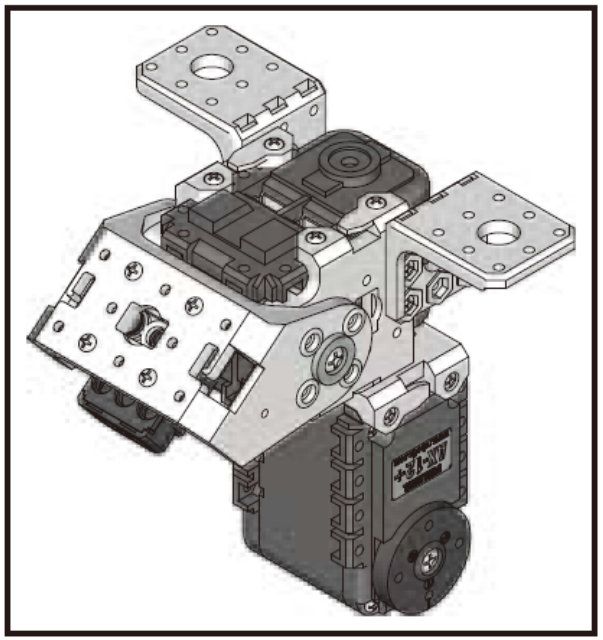


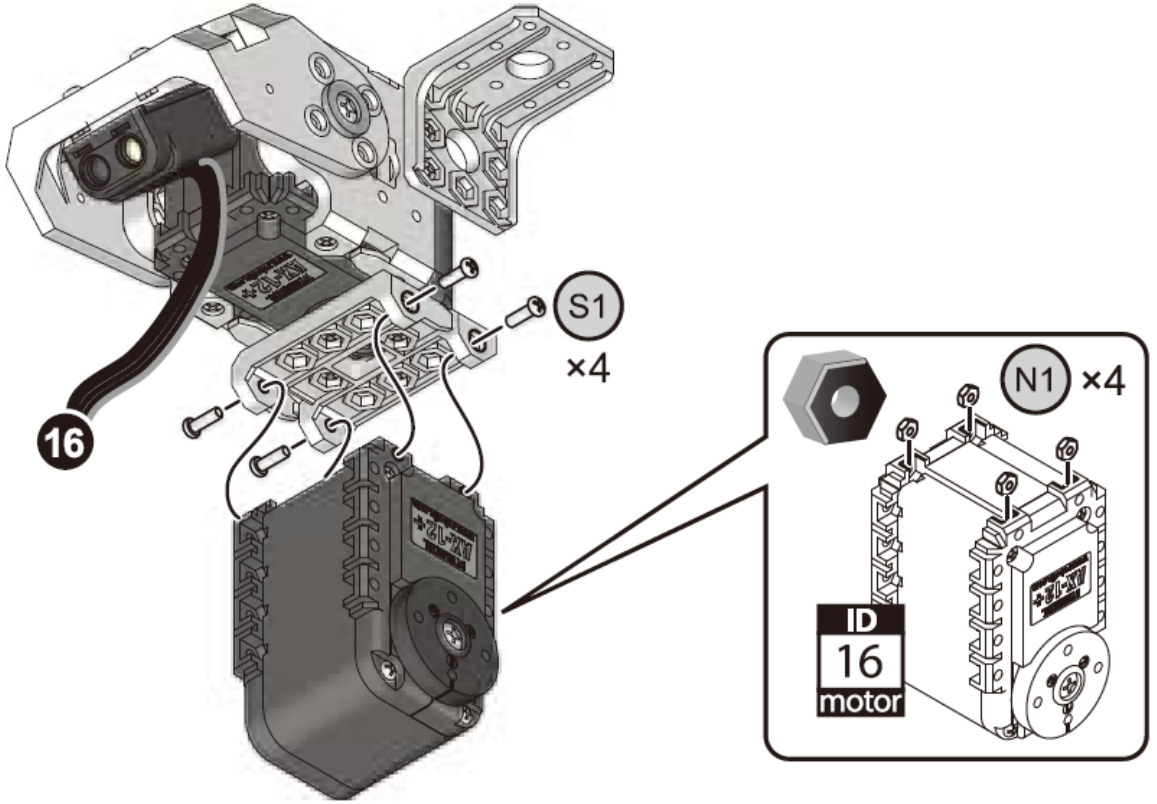
Toto je označení kabelu, kterým lze připojit IR senzor.

Krok 3

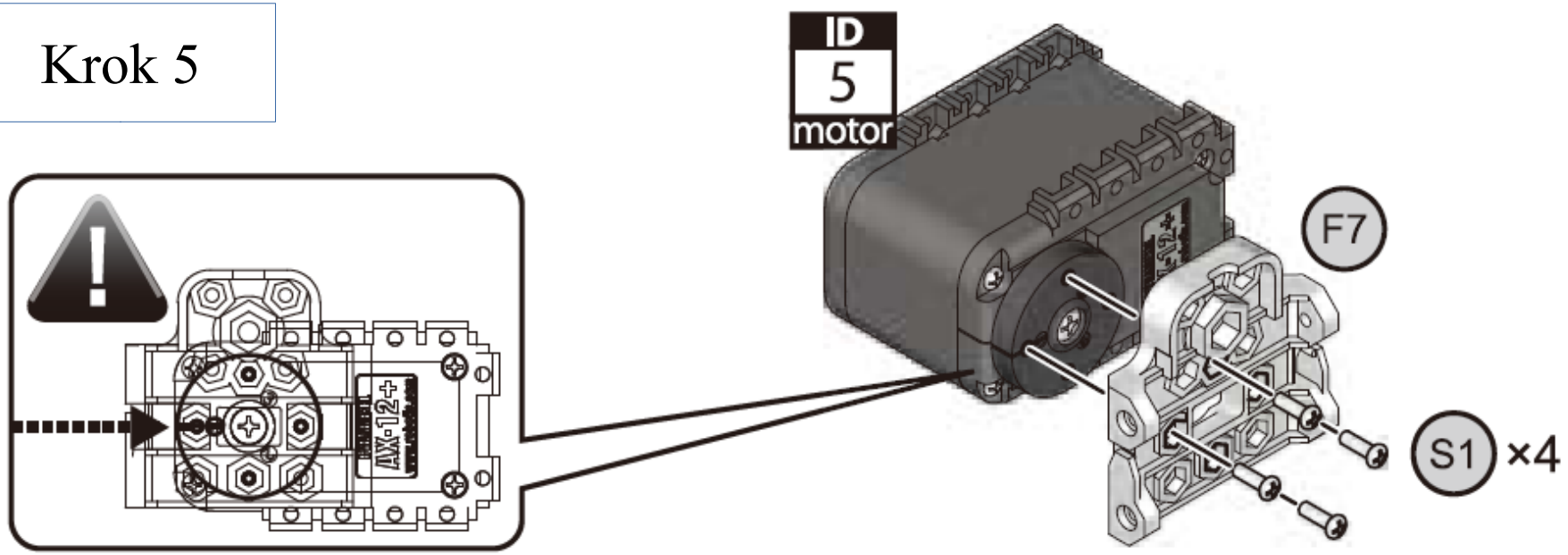


Krok 4

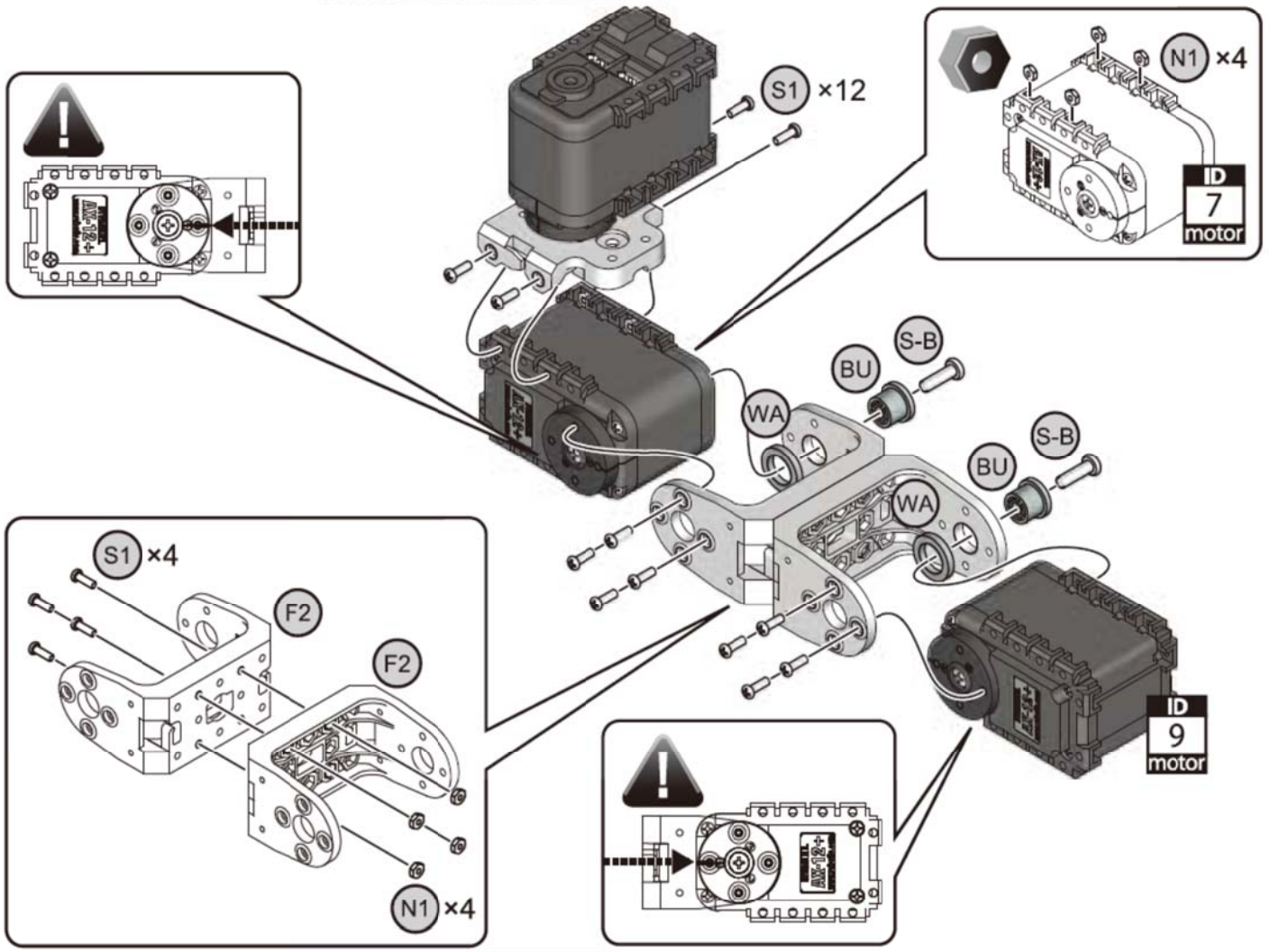




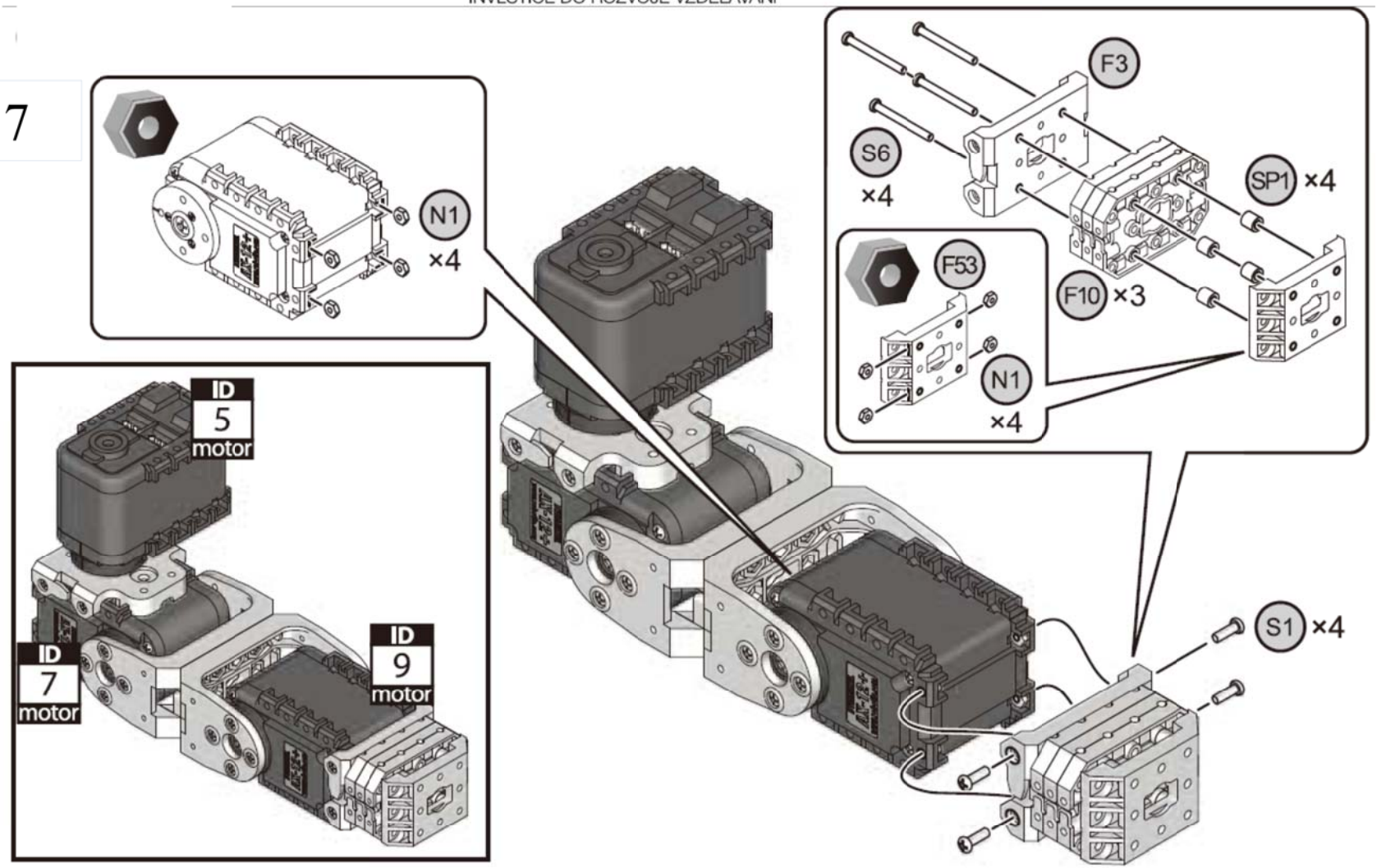
Krok 5



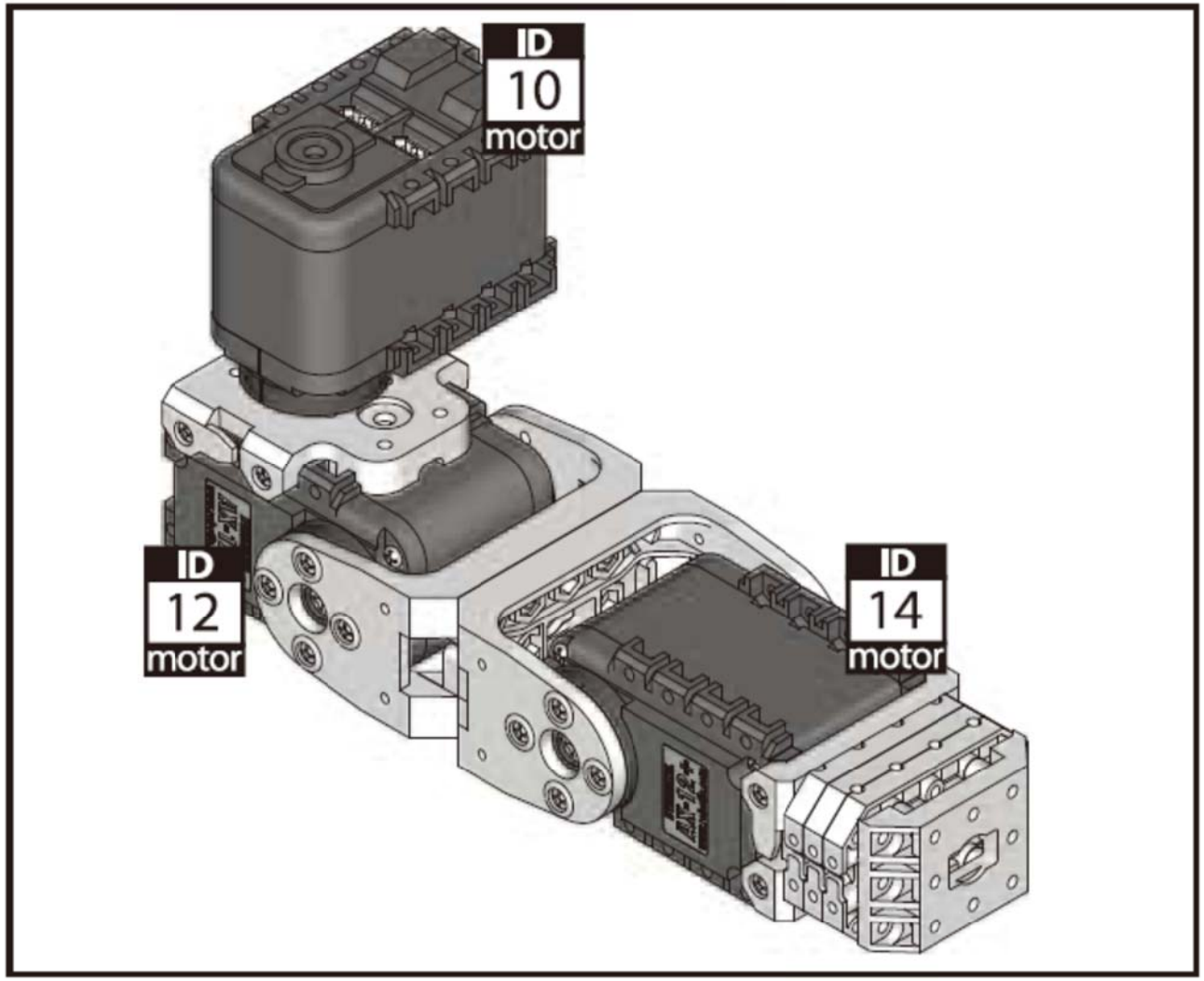
Krok 6



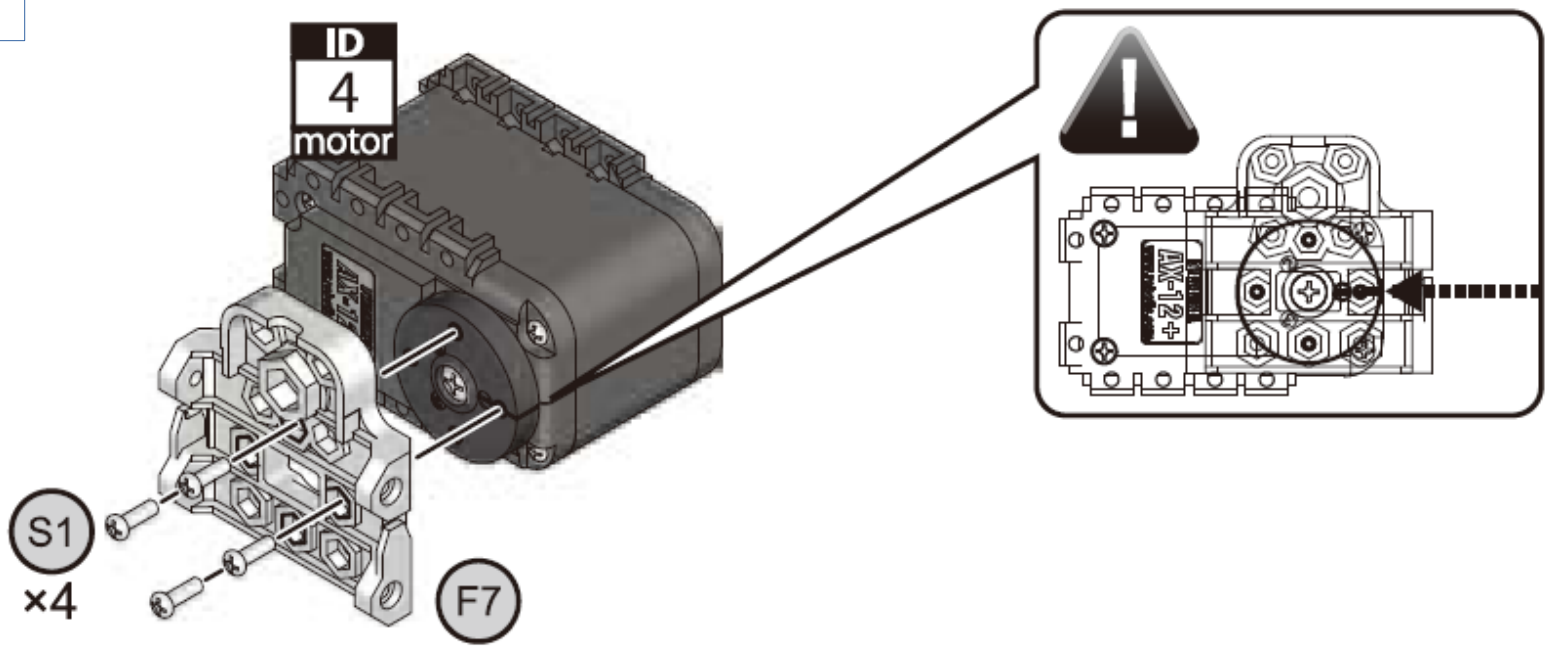
Krok 7



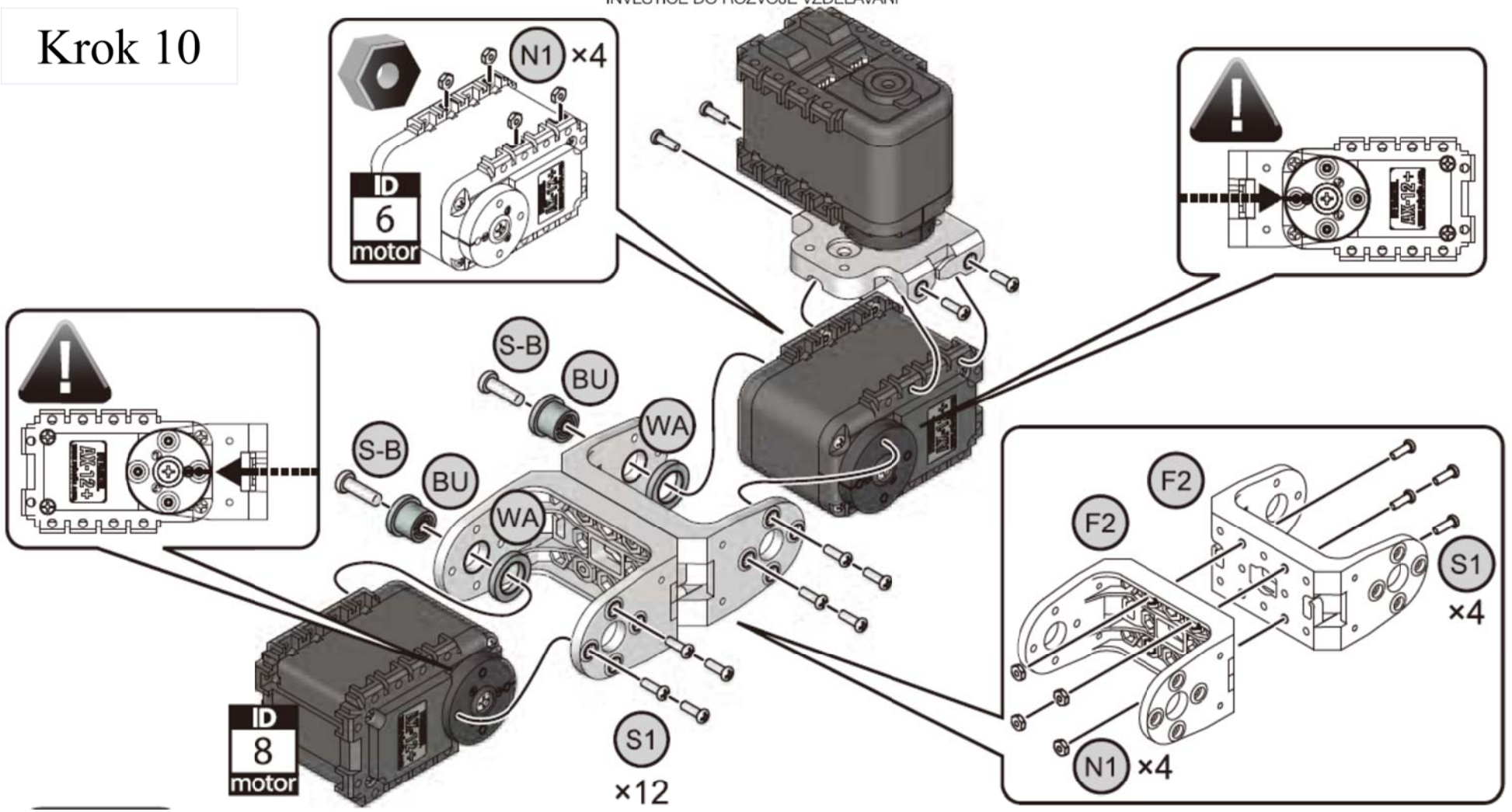
Krok 8



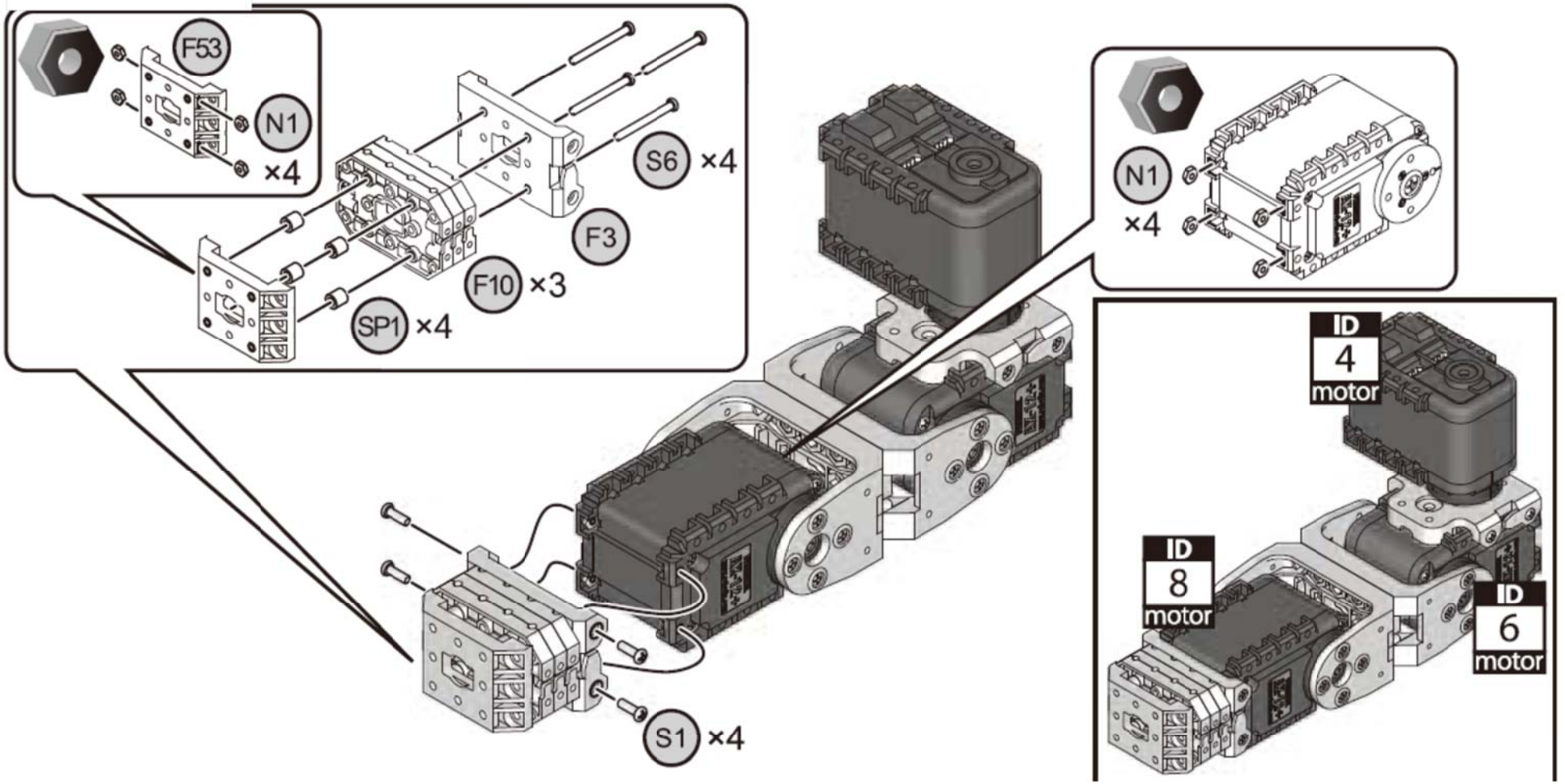
Krok 9



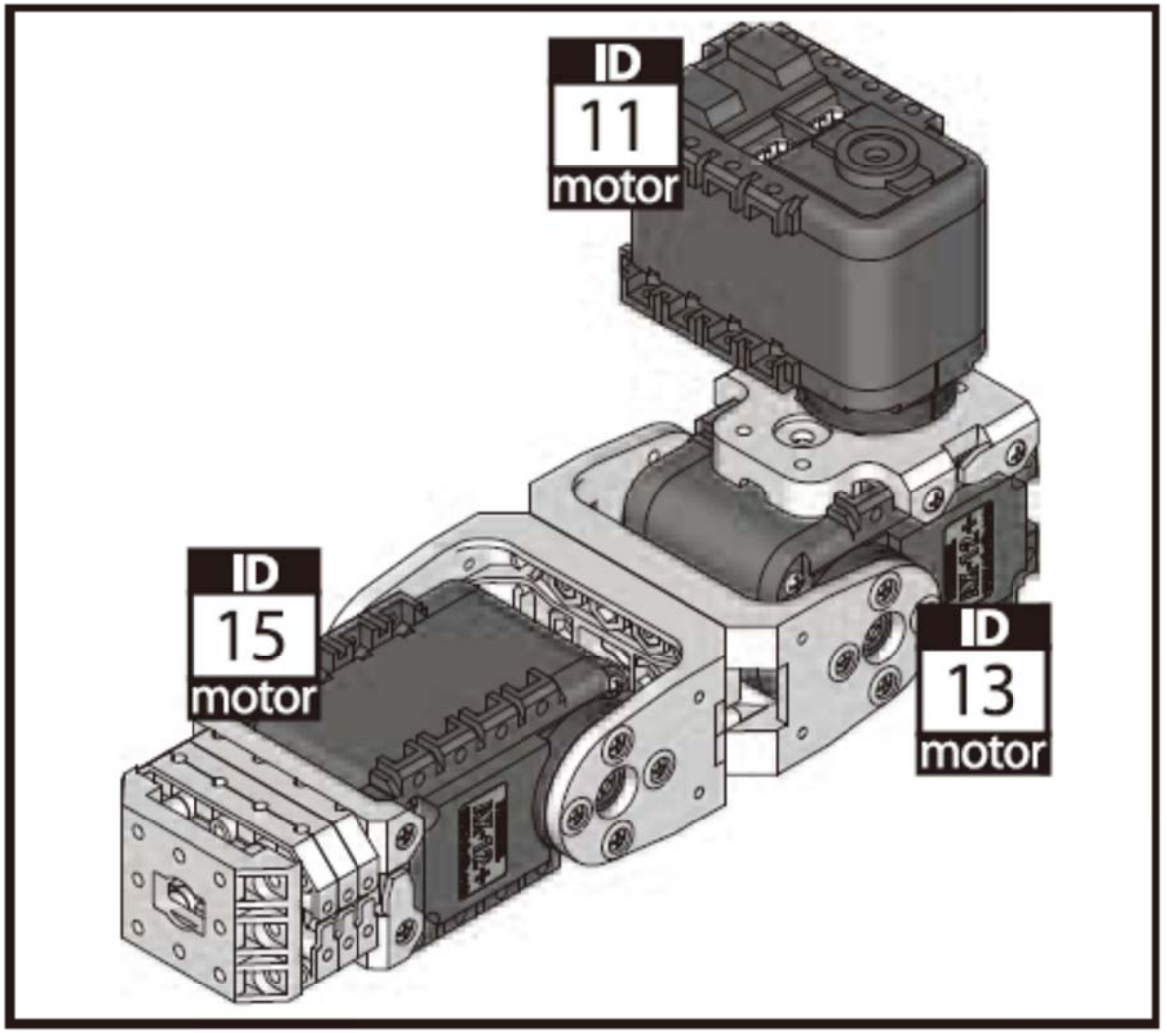
Krok 10



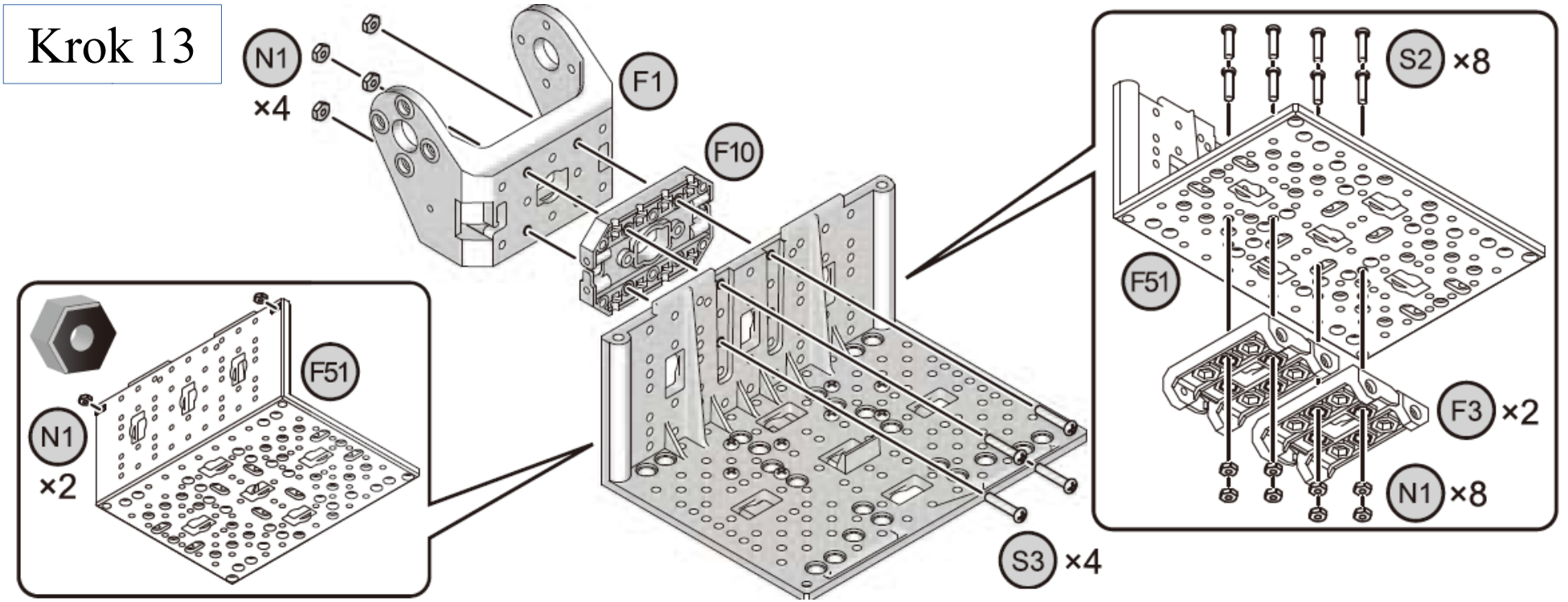
Krok 11



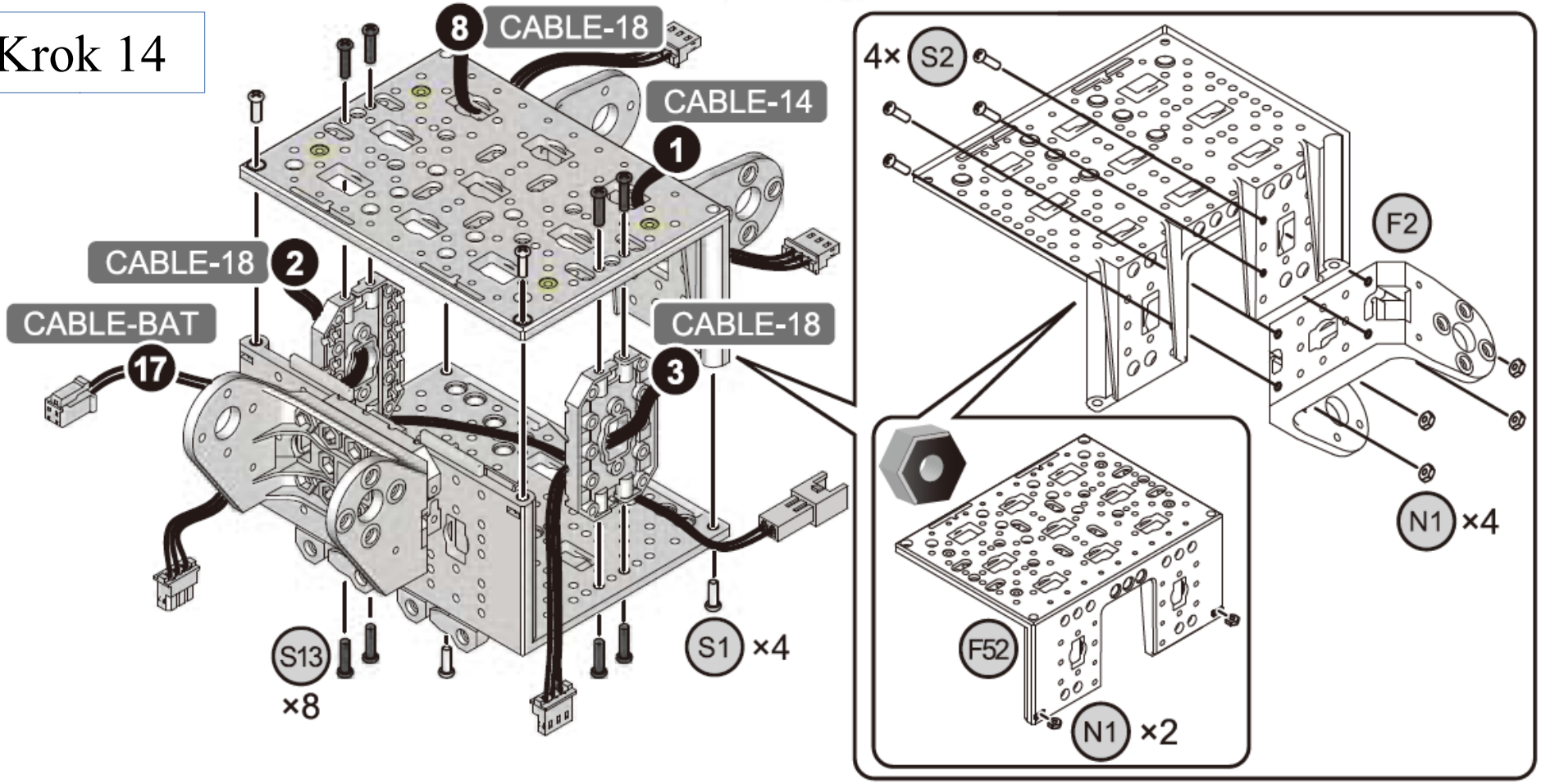
Krok 12



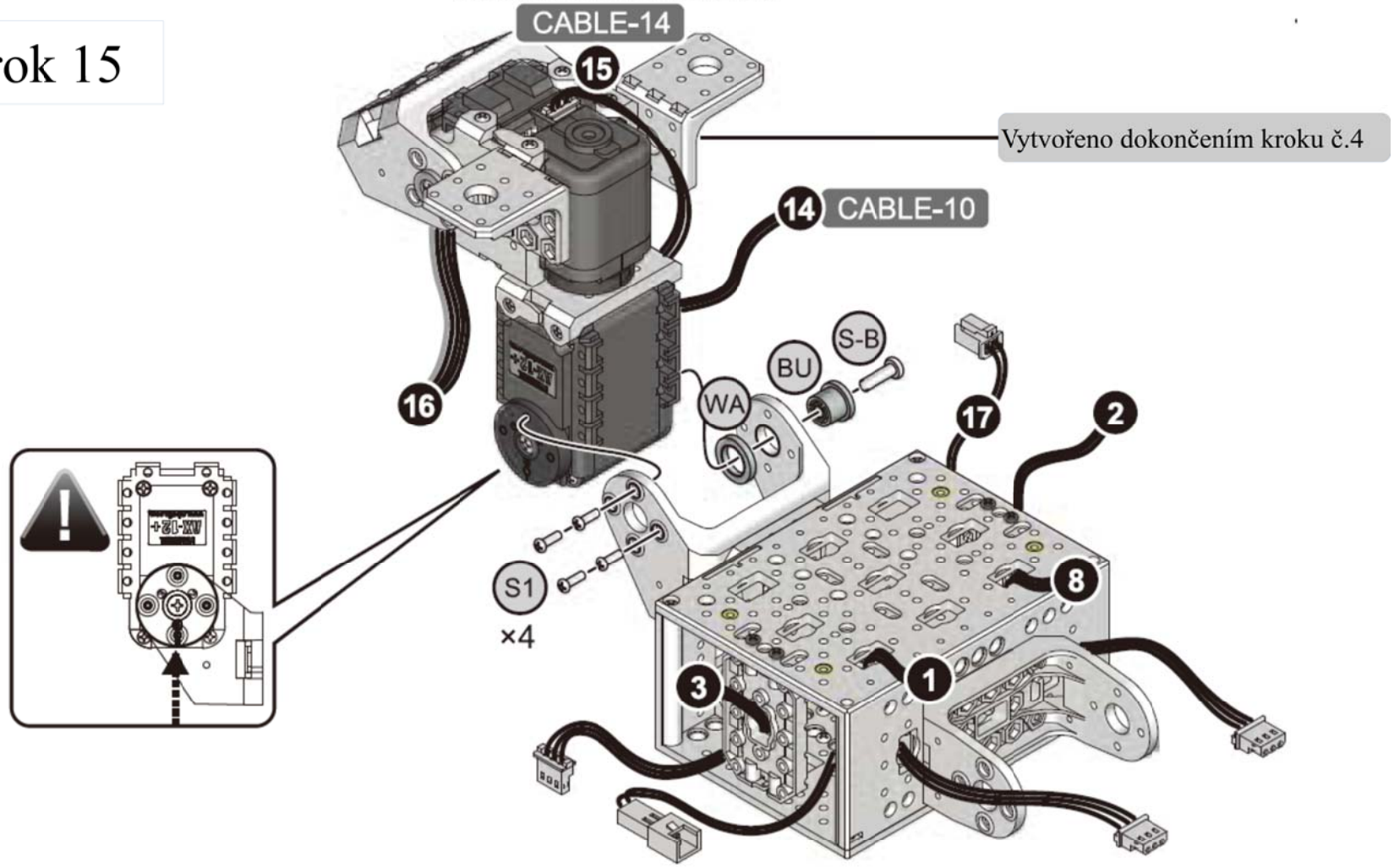
Krok 13



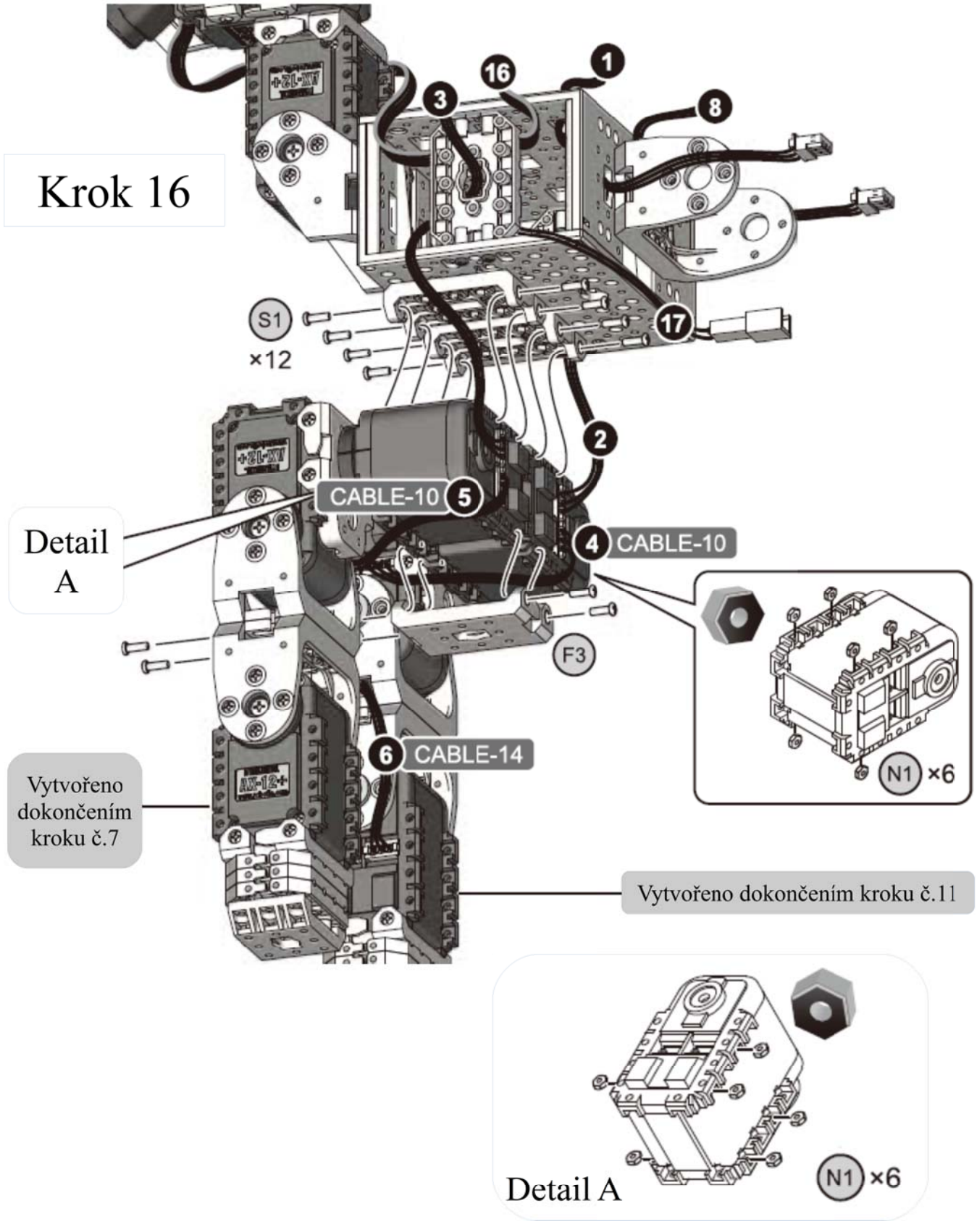
Krok 14


8 CABLE-18
CABLE-14
1
CABLE-18 2
CABLE-BAT
17
CABLE-18
3
S13
×8
S1 ×4
4× S2
F2
N1 ×4
F52
N1 ×2

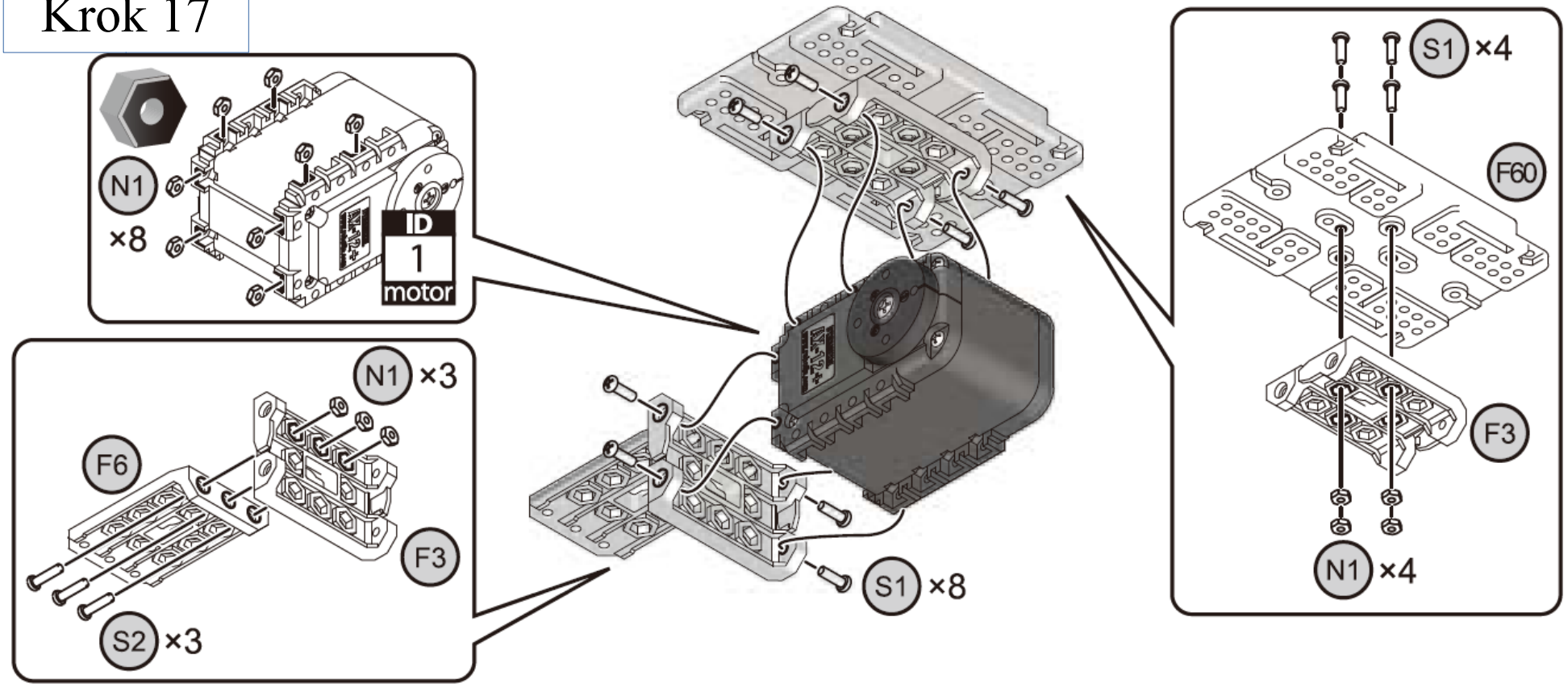
Krok 15


CABLE-14
15
Vytvořeno dokončením kroku č.4
14
CABLE-10
16
BU
S-B
WA
17
2
8
S1
×4
3
1

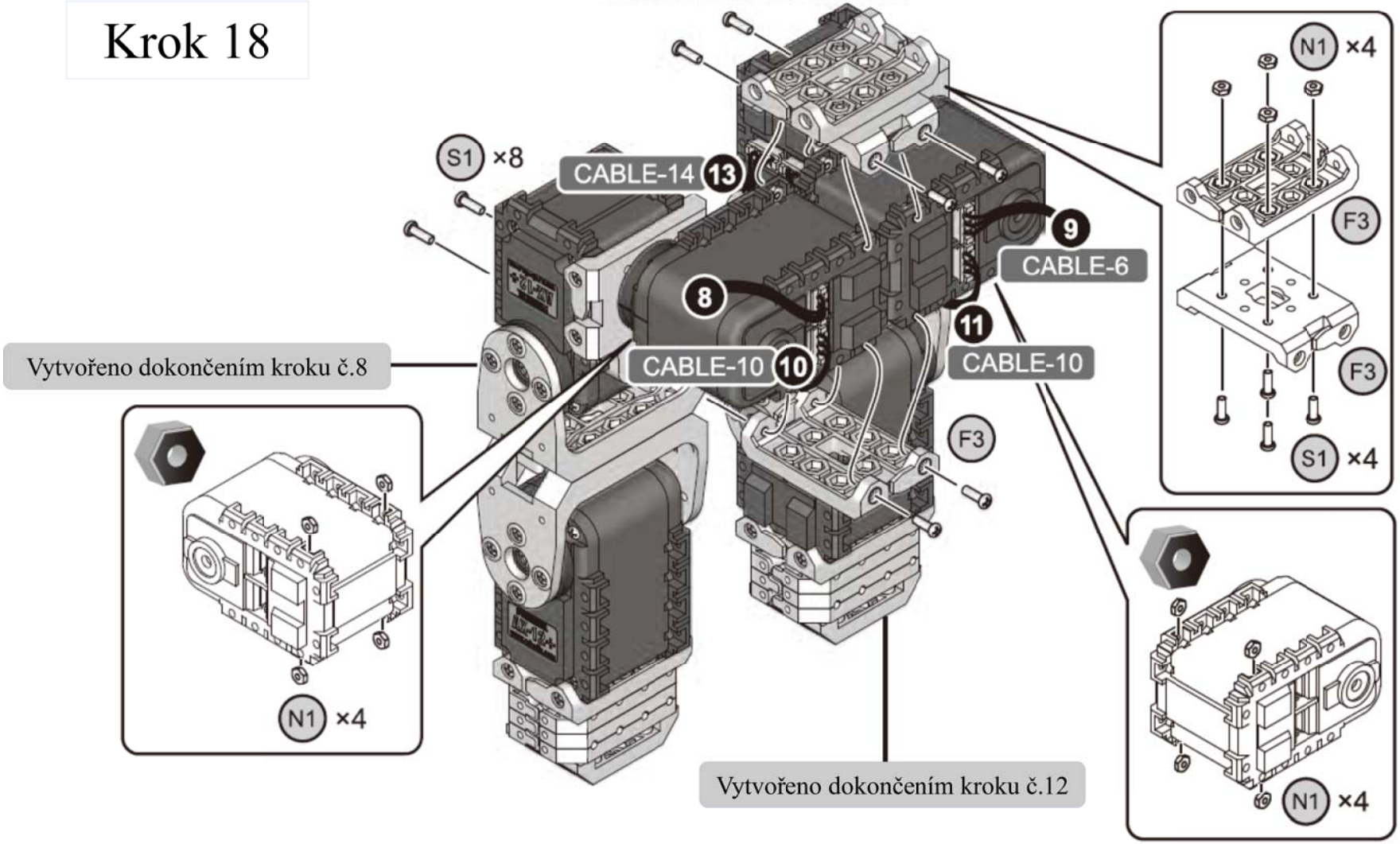
Krok 16


16
1
3
8
S1
×12
17
2
CABLE-10
5
Detail
A
4
CABLE-10
F3
N1
×6
6
CABLE-14
Vytvořeno
dokončením
kroku č.7
Vytvořeno dokončením kroku č.11
Detail A
N1
×6

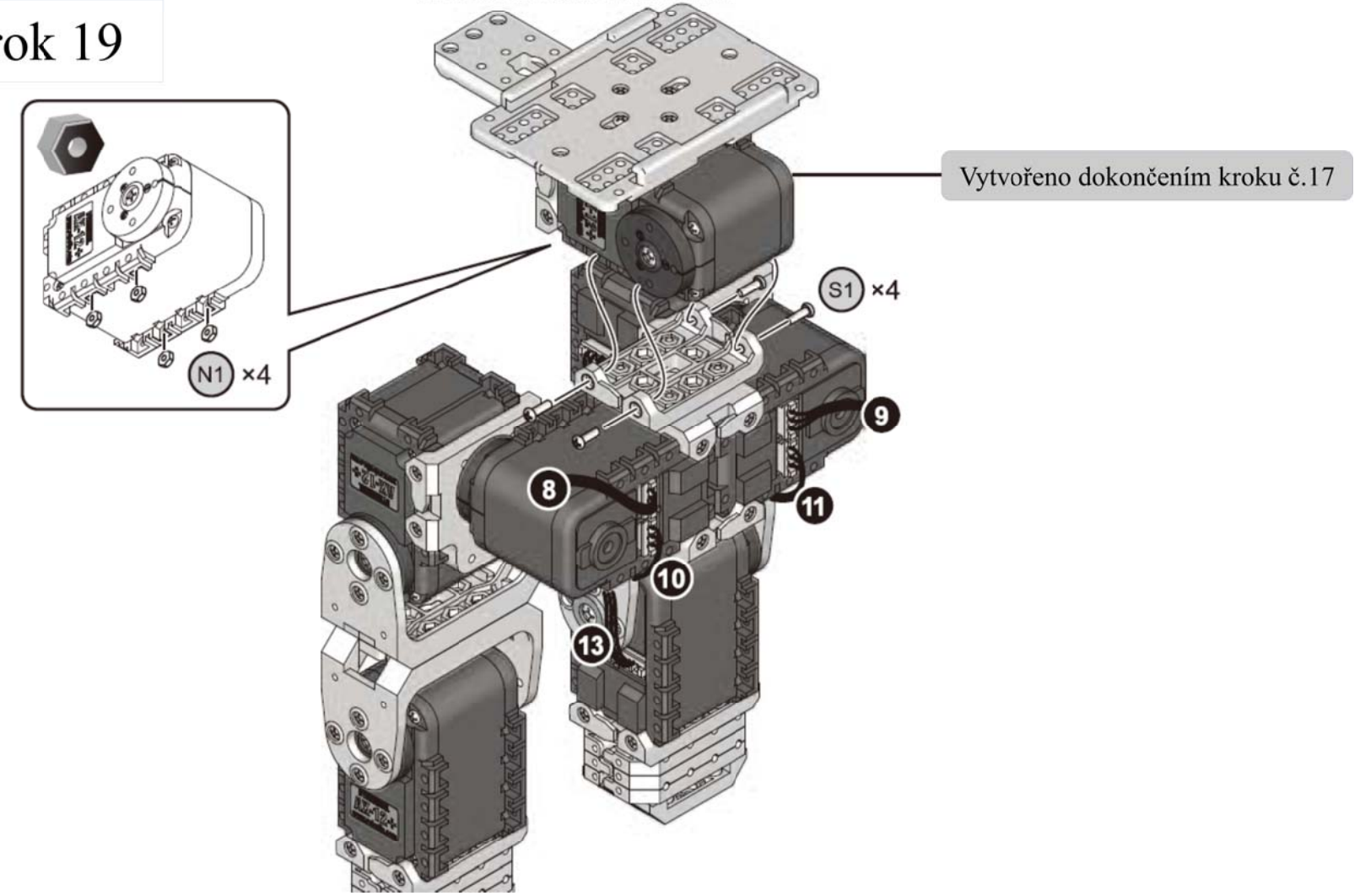
Krok 17



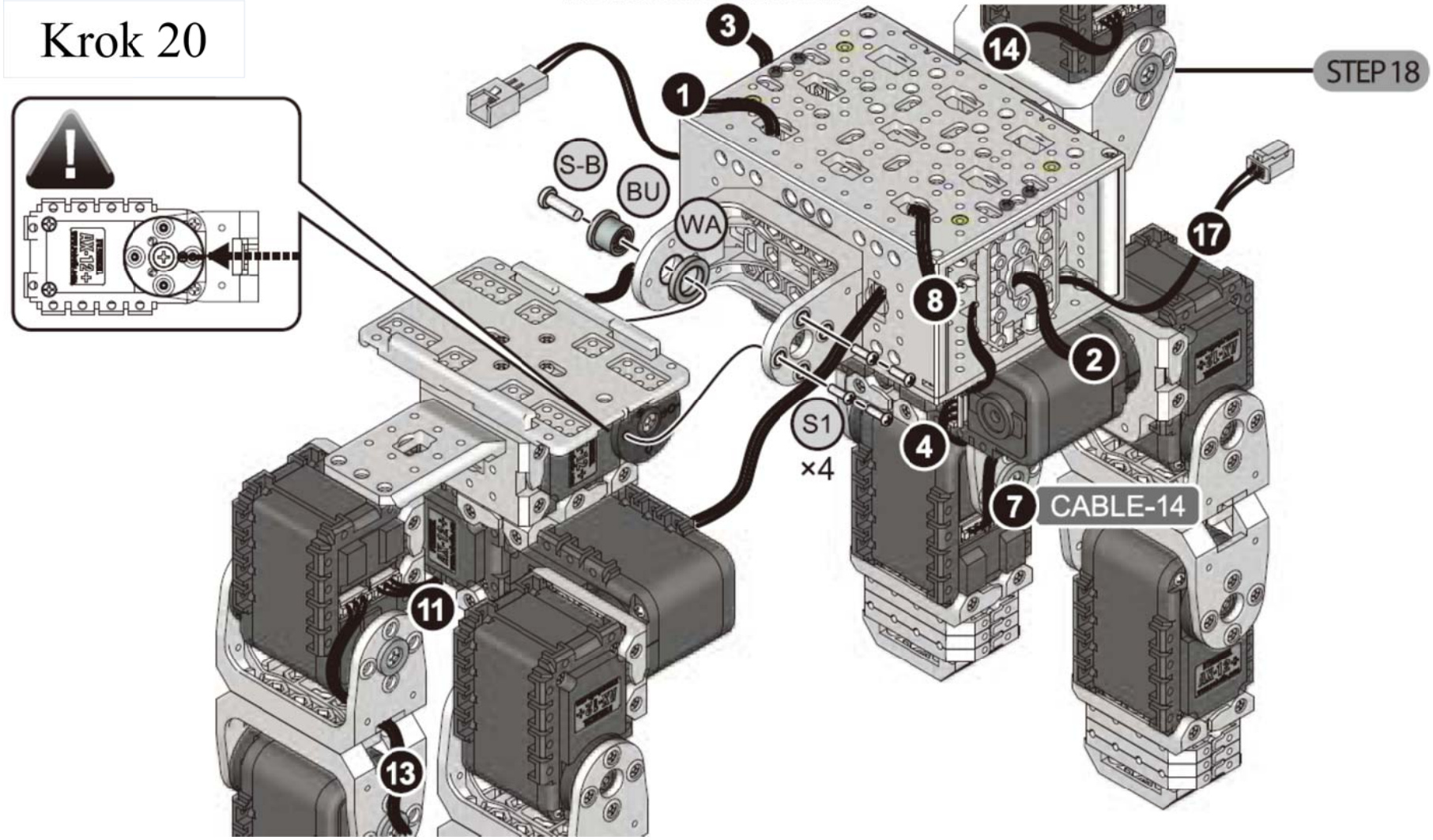
Krok 18


S1 ×8
CABLE-14 13
9
CABLE-6
8
11
CABLE-10
CABLE-10 10
F3
N1 ×4
F3
F3
S1 ×4
Vytvořeno dokončením kroku č.8
N1 ×4
Vytvořeno dokončením kroku č.12
N1 ×4

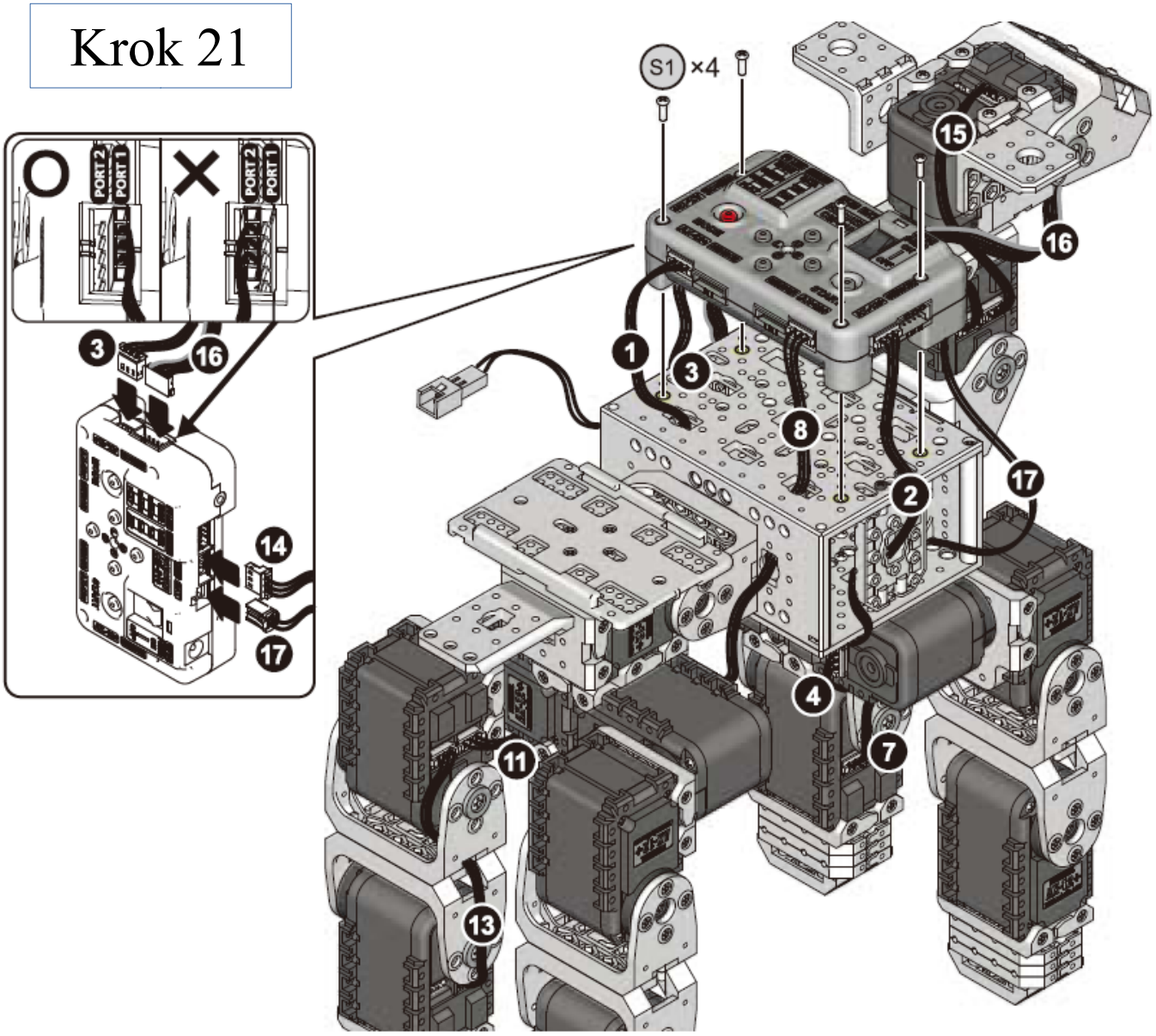
Krok 19



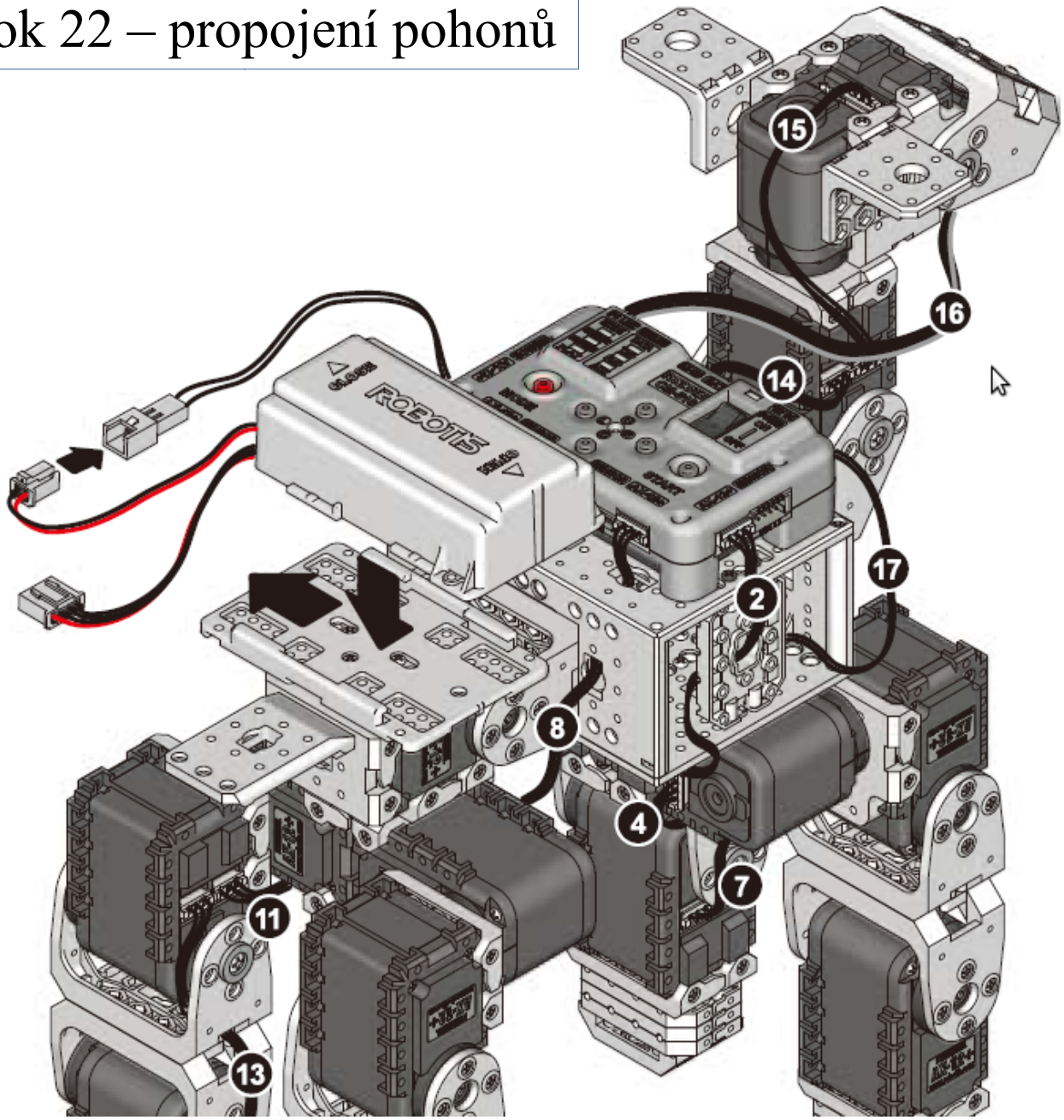
Krok 20

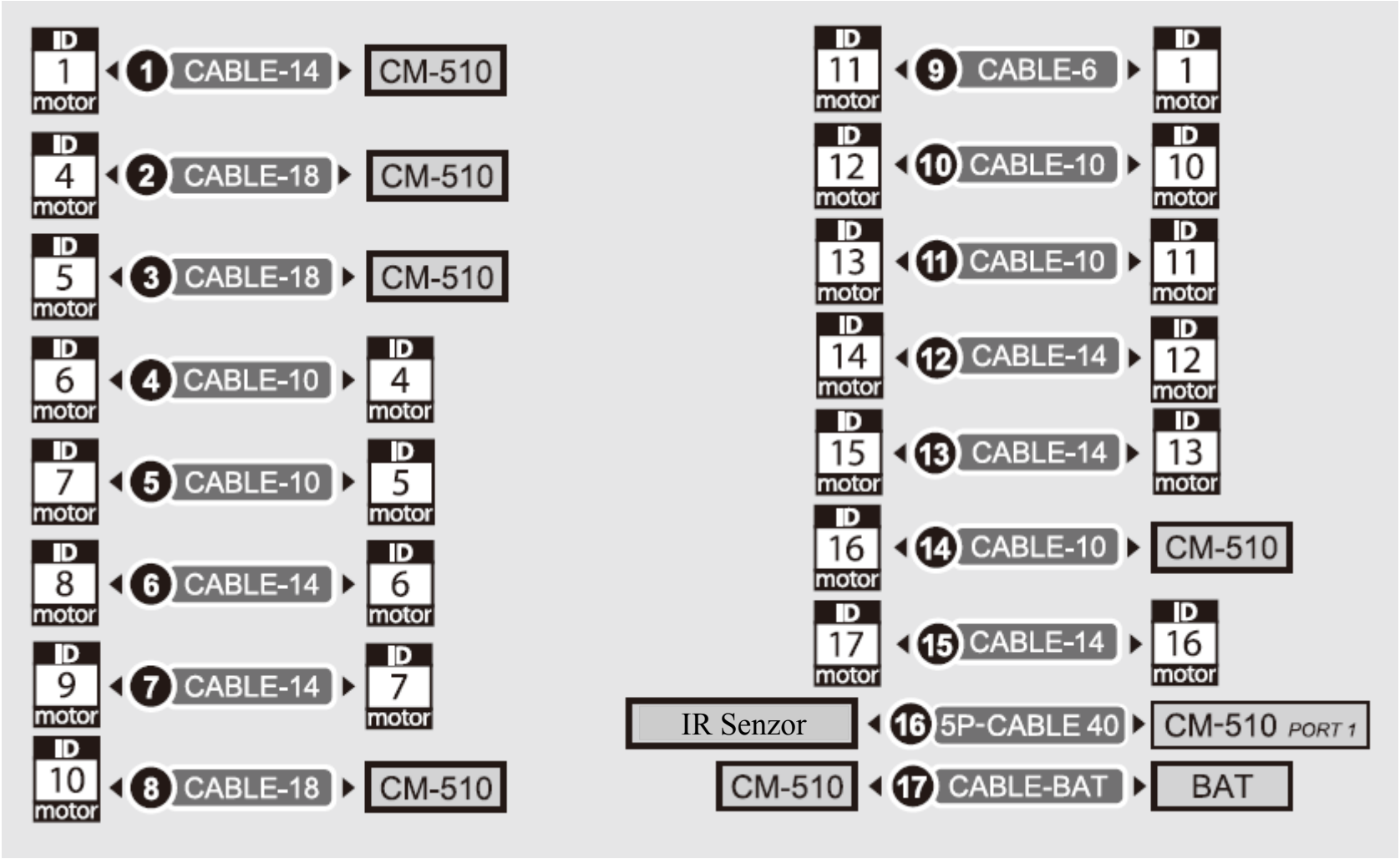


Krok 21

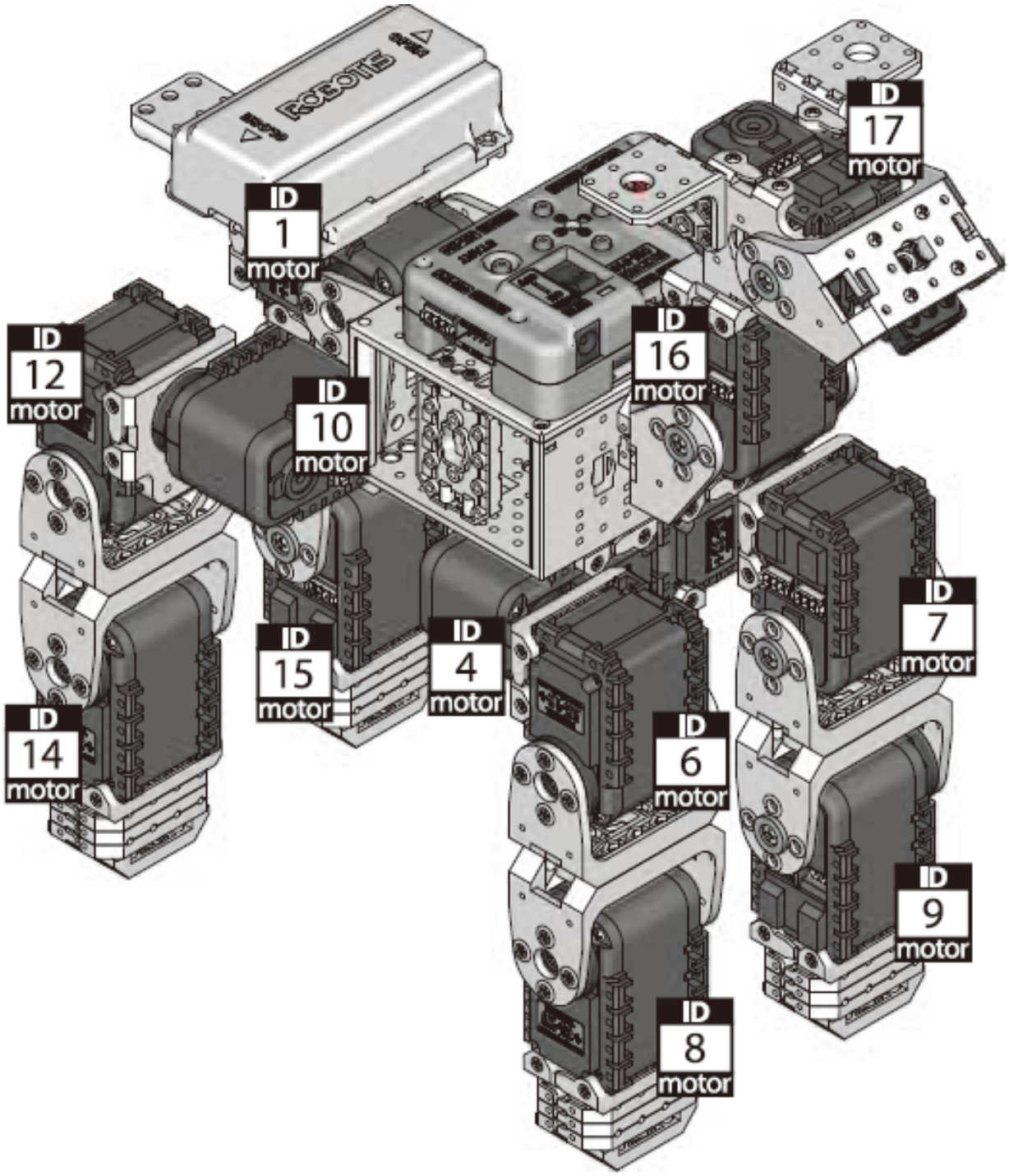


Krok 22 – propojení pohonů

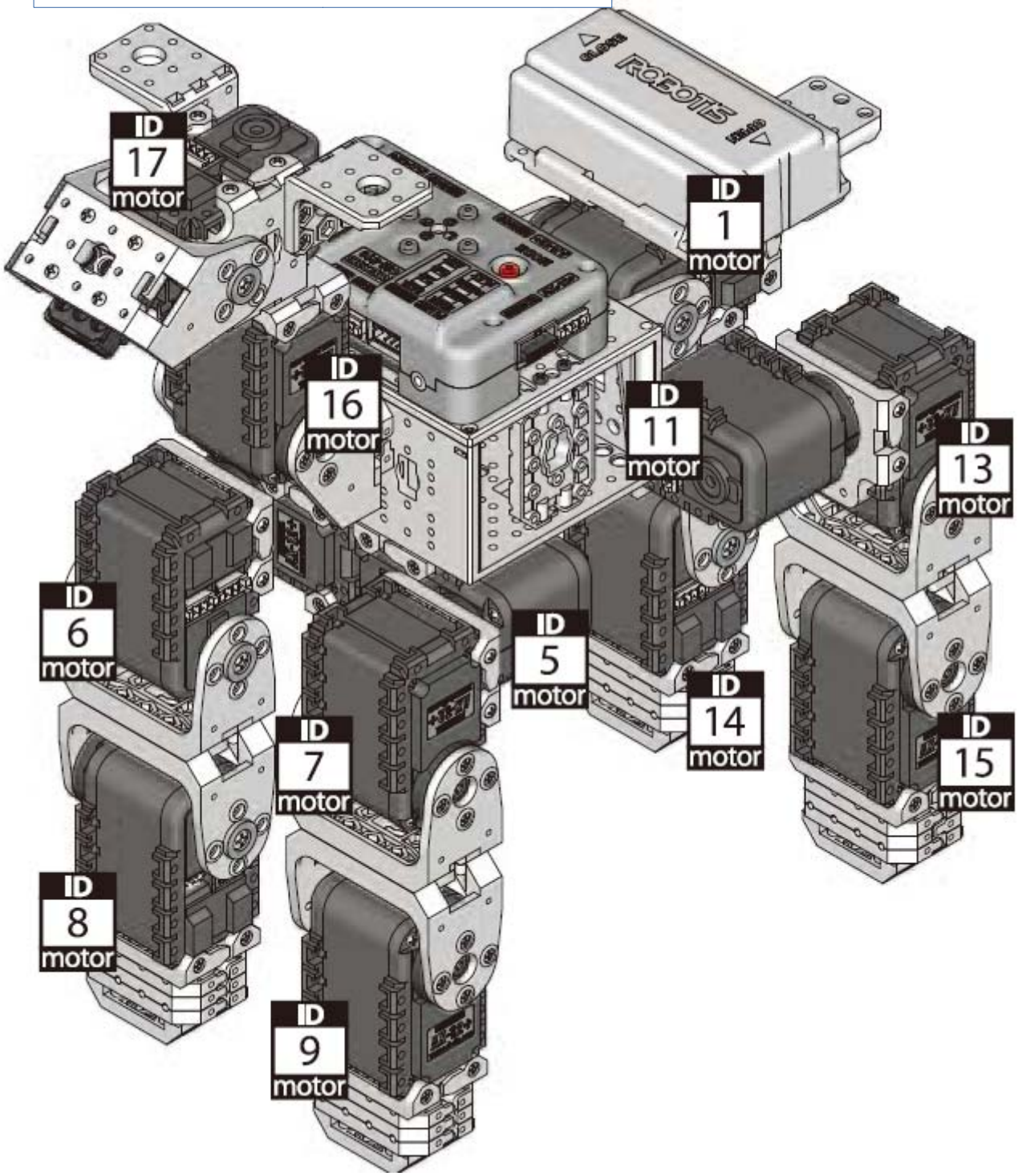


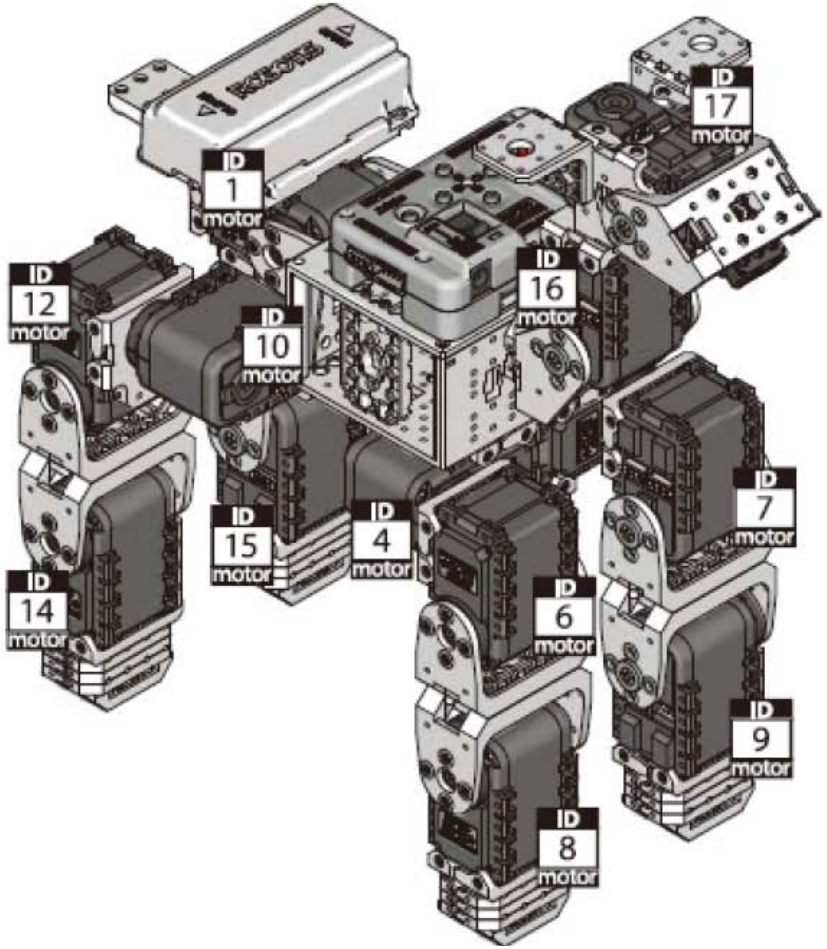

ID 1 motor
1 CABLE-14
CM-510
ID 4 motor
2 CABLE-18
CM-510
ID 5 motor
3 CABLE-18
CM-510
ID 6 motor
4 CABLE-10
ID 4 motor
ID 7 motor
5 CABLE-10
ID 5 motor
ID 8 motor
6 CABLE-14
ID 6 motor
ID 9 motor
7 CABLE-14
ID 7 motor
ID 10 motor
8 CABLE-18
CM-510
ID 11 motor
9 CABLE-6
ID 1 motor
ID 12 motor
10 CABLE-10
ID 10 motor
ID 13 motor
11 CABLE-10
ID 11 motor
ID 14 motor
12 CABLE-14
ID 12 motor
ID 15 motor
13 CABLE-14
ID 13 motor
ID 16 motor
14 CABLE-10
CM-510
ID 17 motor
15 CABLE-14
ID 16 motor
IR Senzor
16 5P-CABLE 40
CM-510 PORT 1
CM-510
17 CABLE-BAT
BAT

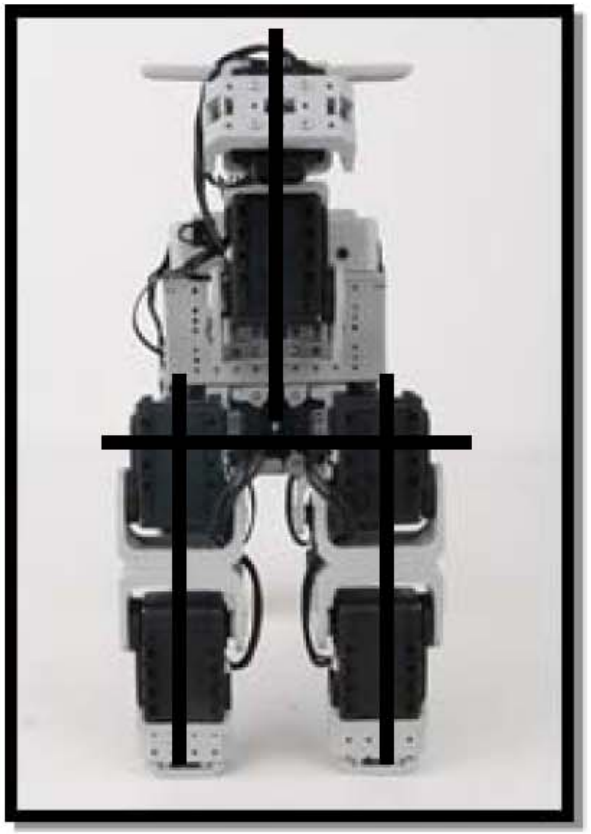
Kontrola identifikátorů

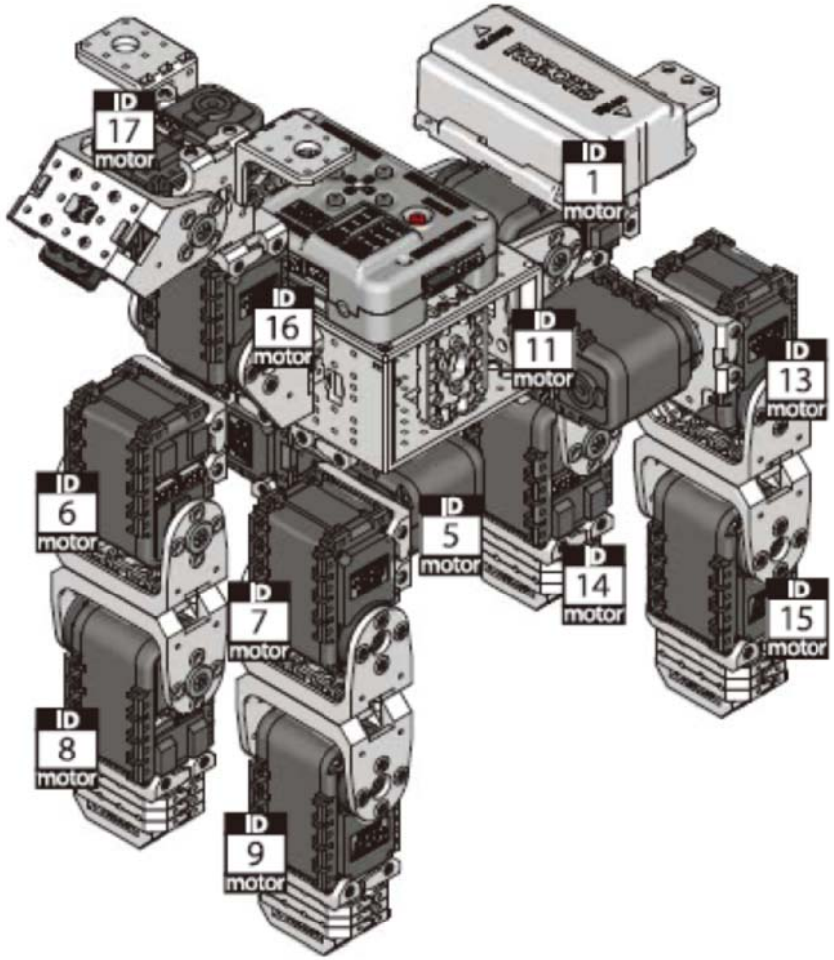


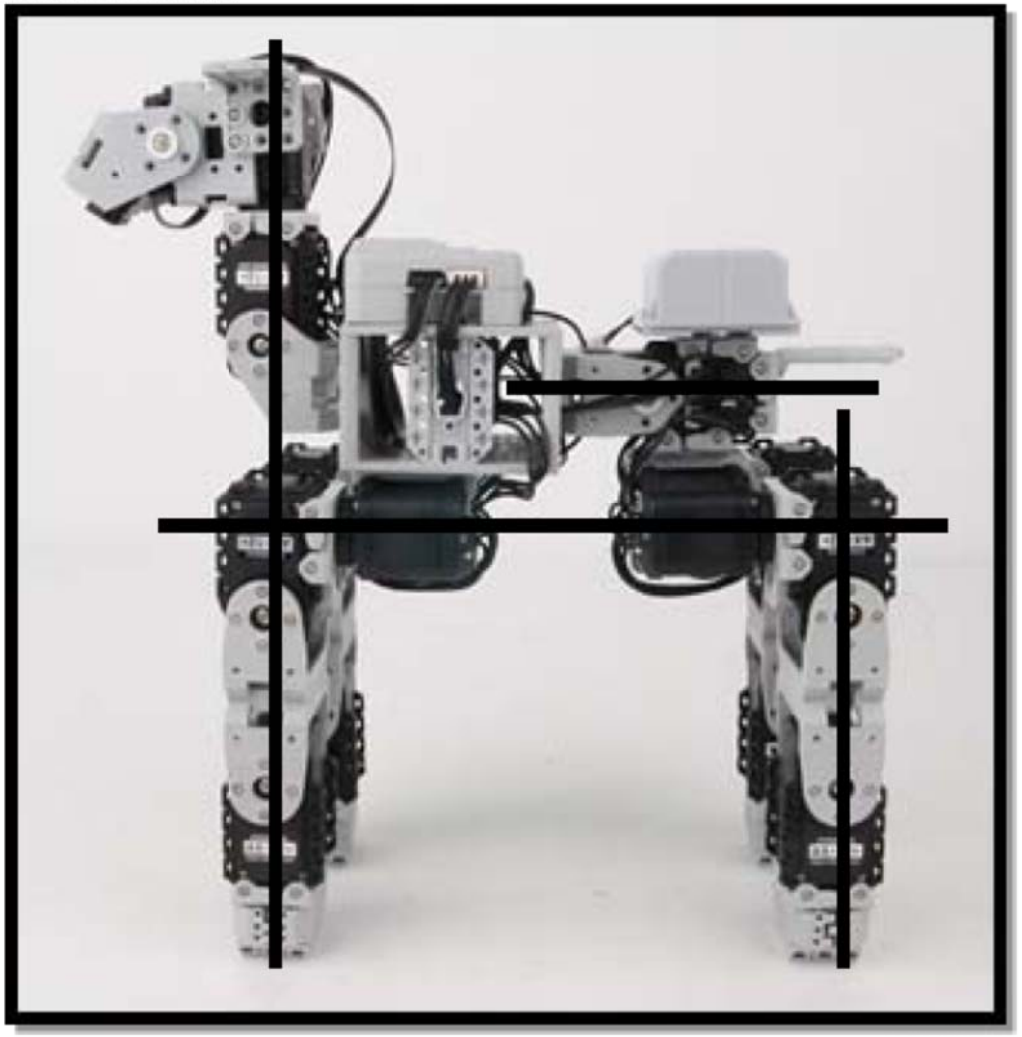
Kontrola identifikátorů




ID 1 motor
ID 17 motor
ID 12 motor
ID 10 motor
ID 16 motor
ID 15 motor
ID 4 motor
ID 7 motor
ID 14 motor
ID 6 motor
ID 9 motor
ID 8 motor




ID 17 motor
ID 1 motor
ID 16 motor
ID 11 motor
ID 13 motor
ID 6 motor
ID 5 motor
ID 14 motor
ID 7 motor
ID 15 motor
ID 8 motor
ID 9 motor



Použitá literatura

[1] ROBOTIS: *Dynamixel AX-12*. User's manual, Korea 2006

[2] ROBOTIS: *Dynamixel Sensor Module AX-S1*. User's manual, Korea 2006

[3] ROBOTIS: *USB2Dynamixel*. User's manual, Korea 2006, Version 1.2

[4] ROBOTIS: *BIO_PRM_Humanoid_ASM_Premium.*. User's manual, Korea 2010

[5] ROBOTIS: *BIO_PRM_Puppy_ASM_EN_Premium.*. User's manual, Korea 2010

[6] ROBOTIS: *BIO_PRM_Dinosaur_ASM_EN.*. User's manual, Korea 2010

ROBOTI

VE ŠKOLE PRO PRAKTICKOU VÝUKU, MOTIVACI I ZÁBAVU

CZ.1.07/1.1.24/01.0066

INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

ROBOTI

Ing. Michal ŘEPKA, Ph.D.