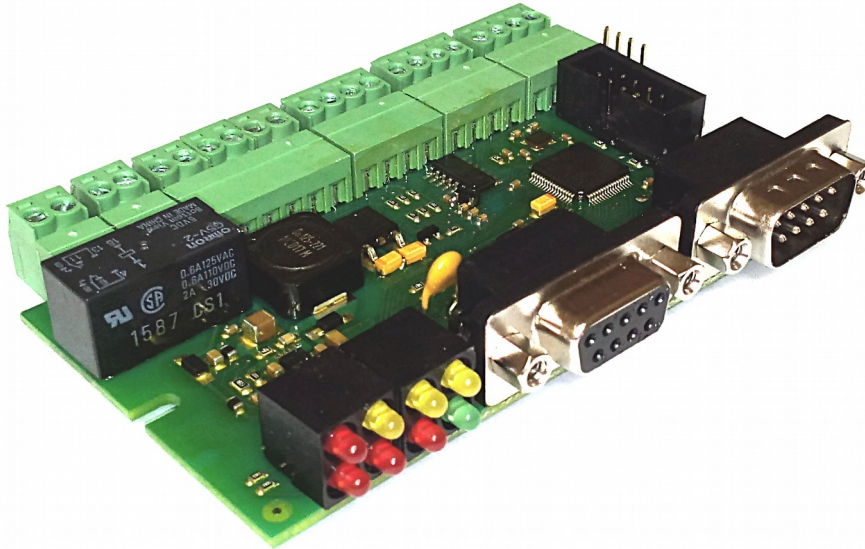




# CNC Technologie a obráběcí stroje

## GVE124 – HW interpolátor

(Plná verze dokumentace v1.0)



### 1 Specifikace

- HW interpolační jednotka s výkonem 50 000/200 000 pulzů/s ve 3-osém pohybu.
- Vnitřní buffer pro 128 vektorů, max. délka vektoru +- 2147483647 kroků
- Řízení krok/směr
- Připojení k PC přes RS232 (USB přes převodník )
- Komunikační rychlost standardně 115 200Bd (lze změnit při objednání, max 921 600Bd)
- 3 vstupy pro referenční spínače
- Konektor pro senzor nástroje nebo obrobkovou sondu
- 1 relé výstup s použitím pro spínání (max 30VDC, 2A) např. vřeteno, laser , odsávání, chlazení atd.
- LED signalizace Power, Data Rx, Data Tx, Run, Rele, RefX, RefY, RefZ
- Napájení 9 – 90VDC
- Odběr max. 170mA (při 12V), 80mA (při 24V)

### 2 Aplikace

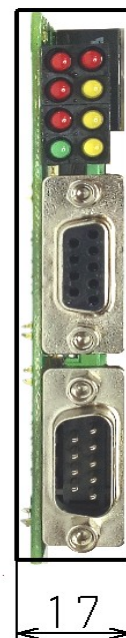
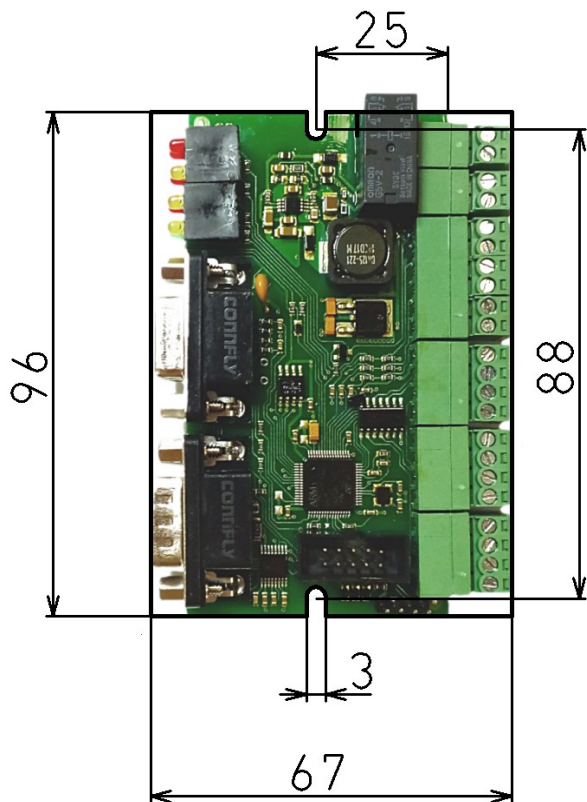
Řízení frézek, gravírek, vrtaček, polohovacích stolů, robotických manipulátorů atd. Ovládání pomocí sw Armote v1.98 a vyšší (frézka/gravírka/vrtačka), nebo vlastní uživatelskou aplikací (ovládání jednotky je popsáno v kapitole 12)

### 3 Součást dodávky

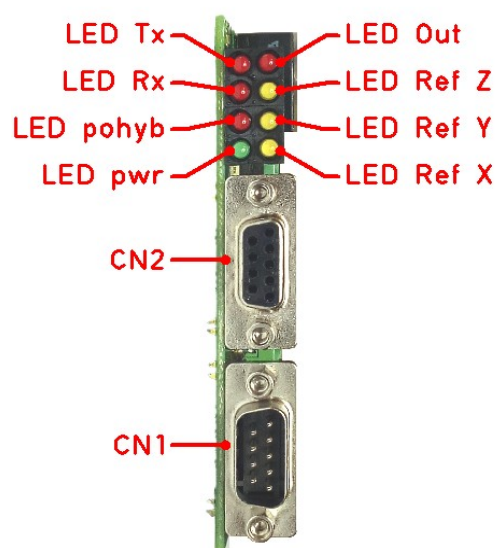
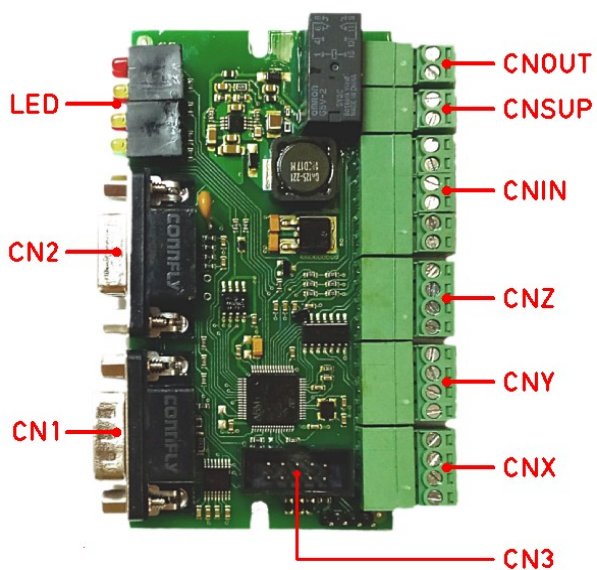
Jednotka GVE124 s protikusy svorkových konektorů

### 4 Rozměry

67 x 96 x 17mm



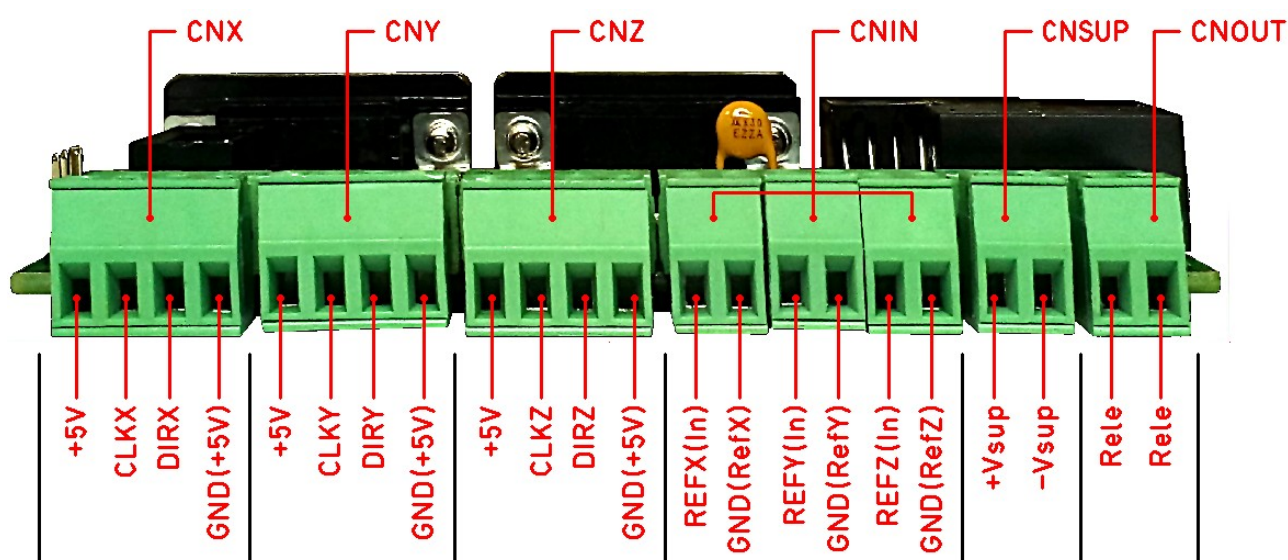
### 5 Přehled



## 6 Popis konektorů

CNX	výstup pro driver osy X	CNOUT	relé výstup (max 30VDC, 2A)
CNY	výstup pro driver osy Y	CNSUP	Napájení (9-90VDC)
CNZ	výstup pro driver osy Z	CN1	Připojení k PC (RS232)
CNIN	vstupy	CN2	Senzor nebo sonda
CN3	Konektor pro Freq124		

## 7 Popis vývodů



**!!! Pro správnou funkci při použití ovládacího sw ARMOTE je nutné dodržet následující použití.**

- Vstupy RefX, RefY a RefZ na konektoru CNIN jsou vyhrazeny pro referenční (home) spínače.
- Výstup Relé na konektoru CNOUT je potřeba nastavit pomocí utility UniGVE config, kterou naleznete na [www.gravos.cz](http://www.gravos.cz) v sekci Ke stažení (výchozí nastavení pro použití spínání vřetene).

konektor	vývod	popis
<b>CNX</b>	+5V	Výstup 5V pro optočlen driveru
	CLKX	Signál KROK (step) pro driver osy X
	DIRX	Signál SMĚR (dir) pro driver osy X
	GND	Napájení - zem
<b>CNY</b>	+5V	Výstup 5V pro optočlen driveru
	CLKY	Signál KROK (step) pro driver osy Y
	DIRY	Signál SMĚR (dir) pro driver osy Y
	GND	Napájení - zem
<b>CNZ</b>	+5V	Výstup 5V pro optočlen driveru
	CLKY	Signál KROK (step) pro driver osy Z
	DIRY	Signál SMĚR (dir) pro driver osy Z
	GND	Napájení - zem
<b>CNIN</b>	REF X	Vstup ref. Spínače osy X, uzemňuje se k GND
	GND	Zem vstupů
	REF Y	Vstup ref. Spínače osy Y, uzemňuje se k GND
	GND	Zem vstupů
	REF Z	Vstup ref. Spínače osy Z, uzemňuje se k GND
	GND	Zem vstupů
<b>CNOUT</b>	relé	Kontakty relé
		Kontakty relé
<b>CNSUP</b>	napájení	Napájení 9-90VDC, 150mA max (při 12VDC) viz. Doporučené zapojení
<b>CN1</b>	RS232	Konektor sériového rozhraní RS232 viz. Doporučené zapojení
<b>CN2</b>		Konektor připojení sondy nebo senzoru
<b>CN3</b>		Konektor pro připojení desky analog výstupu

## 8 Funkce LED signalizace

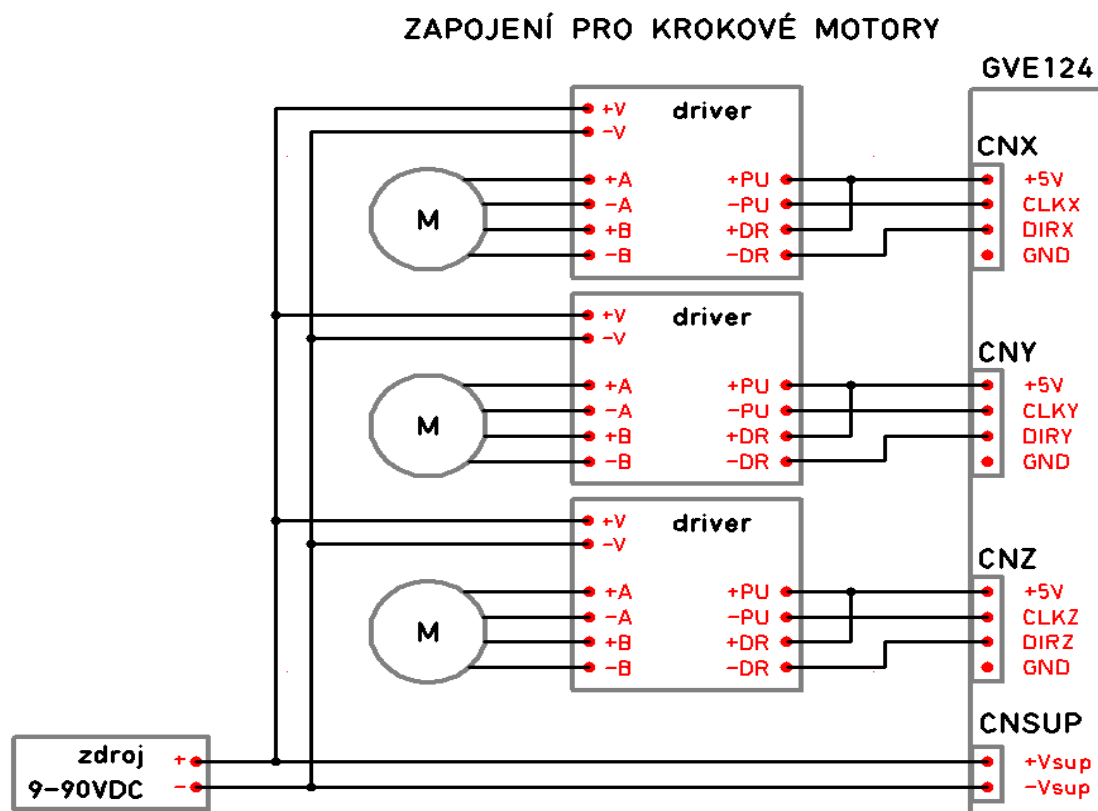
- **PWR** – bliká když je jednotka napájena.
- **Pohyb** – svítí během pohybu (řídící pulzy CLK).
- **Rx** – svítí při příjmu dat.
- **Tx** – svítí při vysílání dat.
- **Out** – svítí při sepnutém relé.
- **RefX** – svítí při sepnutém ref spínači osy X.
- **RefY** – svítí při sepnutém ref spínači osy Y.
- **RefZ** – svítí při sepnutém ref spínači osy Z.

## 9 Příklady doporučeného zapojení

### 9.1 Připojení pohonů zařízení (CNX – CNZ)

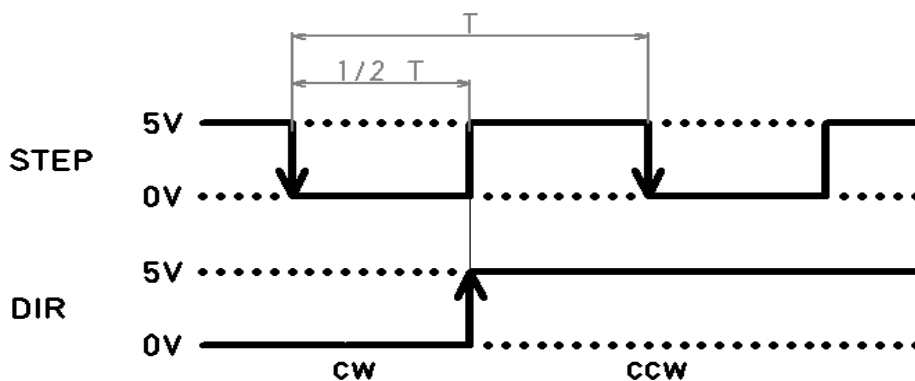
#### 9.1.1 Připojení krokových motorů

(nebo jiných pohonů řízených signály krok/směr)



#### 9.1.2 Časování signálů KROK a SMĚR

Délka pulzu je vždy  $\frac{1}{2}$  periody, při 200kHz je délka pulzu 2,5uS, při 50kHz je délka pulzu 10uS



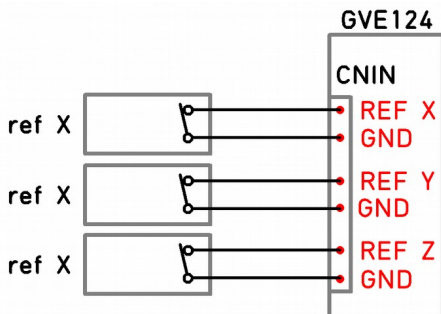
## 9.2 Zapojení vstupů (CNIN)

### 9.2.1 Připojení referenčních spínačů

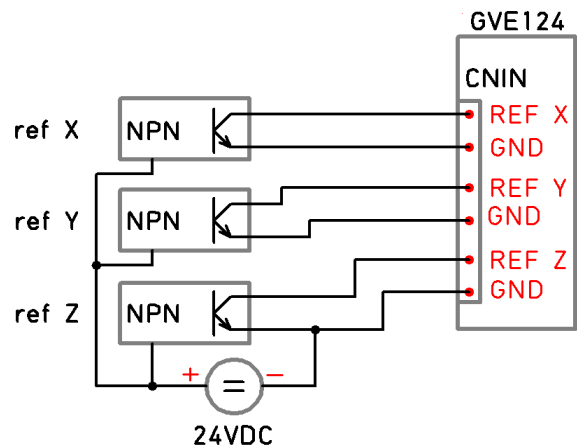
(pro referenční spínače v systémech GRAVOS-ARMOTE)

Ref. spínače doporučujeme rozpínací, aby při poškození kabelu došlo k zastavení stroje.

#### Připojení mechanických spínačů



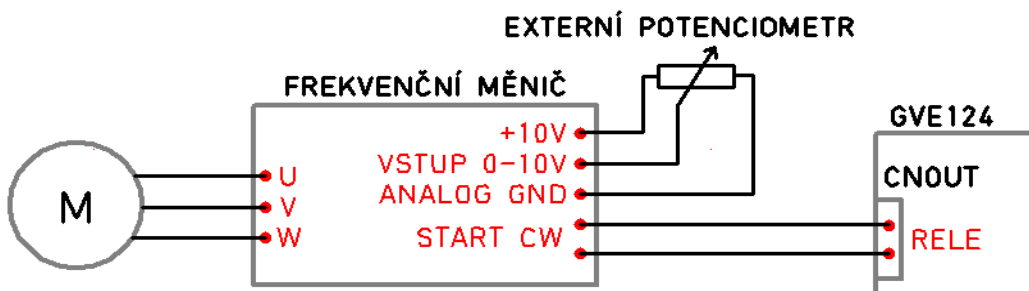
#### Připojení indukčních spínačů



## 9.3 Zapojení výstupu (CNOU)

### 9.3.1 Připojení signálu START k frekvenčnímu měniči

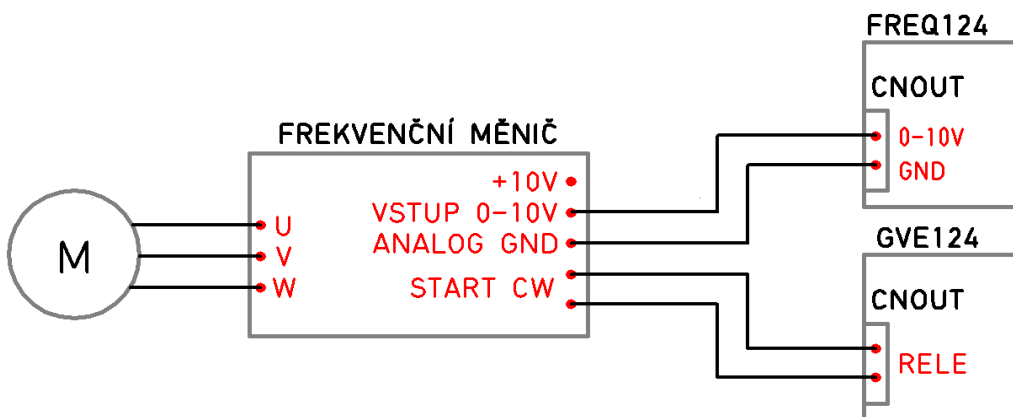
(nastavení otáček se provádí externím potenciometrem nebo na panelu fr. měniče)



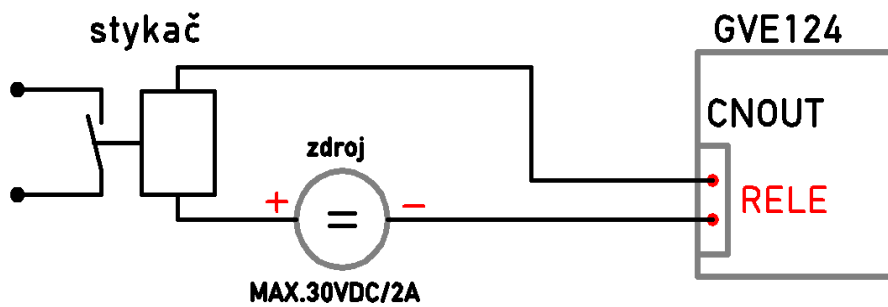
### 9.3.2

#### Připojení signálu START a 0-10V k frekvenčnímu měniči

Toto připojení vyžaduje přídatnou desku Freq pro analogový výstup 0-10v pro nastavení otáček.

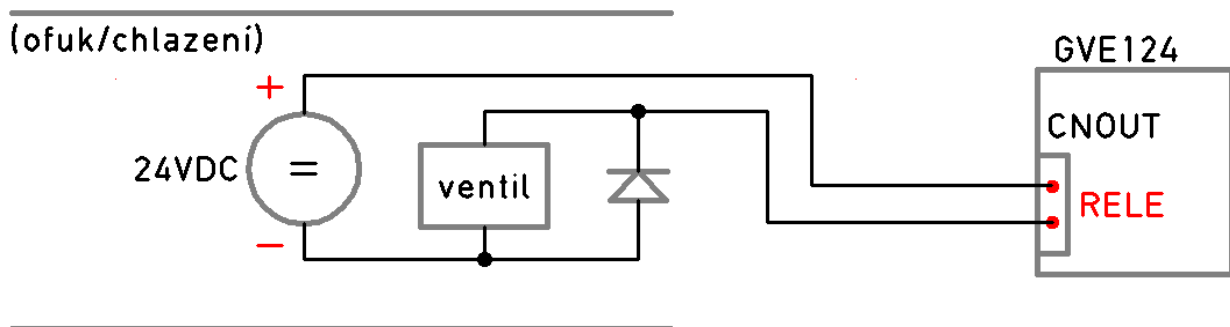


### 9.3.3 Připojení stykače pro spínání větší zátěže než 30VDC/2A



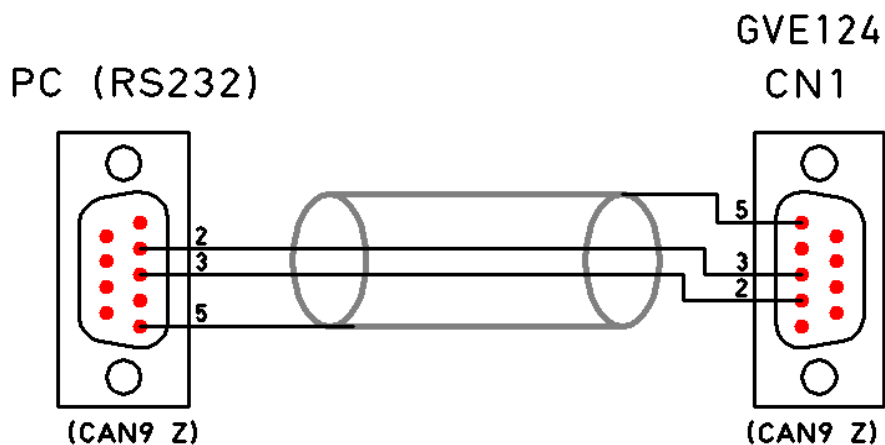
### 9.3.4 Připojení elmag. ventilu

(pro spínání chlazení nebo ofuku nástroje v systémech GRAVOS-ARMOTE, max. zátěž kontaktů relé je 30VDC/2A)



### 9.4 Kabel k připojení GVE124 k PC (CN1)

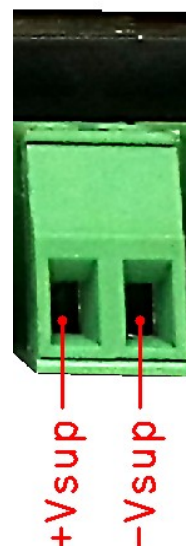
(propojovací kabel je možné objednat pod označením KAB-GVE-A15)



## 9.5 Napájení (CNSUP)

Napájecí napětí  
9 – 90VDC

Díky velkému rozsahu vstupního napájecího napětí lze použít 1 zdroj pro napájení GVE124 i driverů motorů.



## 10 Možné verze jednotky

### Kód pro objednání

jednotka	Intp. Výkon [kHz]	Kom. rychlost [Bd]
GVE124 -	XX	- YY
	50	38400
		57600
	200	115200
		230400
		460800
		921600

Např jednotka GVE124-50-115200, je jednotka s interpolačním výkonem 50kHz a komunikační rychlostí 115200Bd.

Výchozí adresa interpolační jednotky je 0 a adresa pro řízení vřetene je 7. Tyto adresy pokud chcete jiné než výchozí, např při připojení více jednotek GVE na 1 komunikační linku, je potřeba upřesnit v objednávce. Adresy ani komunikační rychlost nelze měnit uživatelsky.



## 11 Nastavení funkce výstupů

Funkce výstupů pro ovládací sw ARMOTE lze konfigurovat pomocí příkazu write (viz kapitola 12.5.9) zapsáním hodnot na příslušné adrese výstupu nebo použít utilitu UniGVE\_config (utilitu lze stáhnout i na [www.gravos.cz](http://www.gravos.cz) v části ke stažení.)

příklad: !0EW01,00 (nastaví Relé pro ovládání vřetene)

### 11.1 Adresy EEPROM pro výstupy

adresa	výstup
0x01	Relé (cnout)
0x05	Výstup signálů STEP/DIR
0xFD	Čas po zapnutí pro odbrzdění brzdy v 0,1s (max 5s)

### 11.2 Hodnoty nastavení pro RELÉ

hodnota	funkce
0x00	vřeteno
0x01	ovládání laseru
0x02	chlazení nástroje
0x03	ofuk nástroje
0x04	zámek krytu stroje
0x05	uvolnění nástroje
0x06	otevření krytu nástrojů
0x08	brzda
0x09	signalizace přerušení
0x10- 0xFF	nepoužito

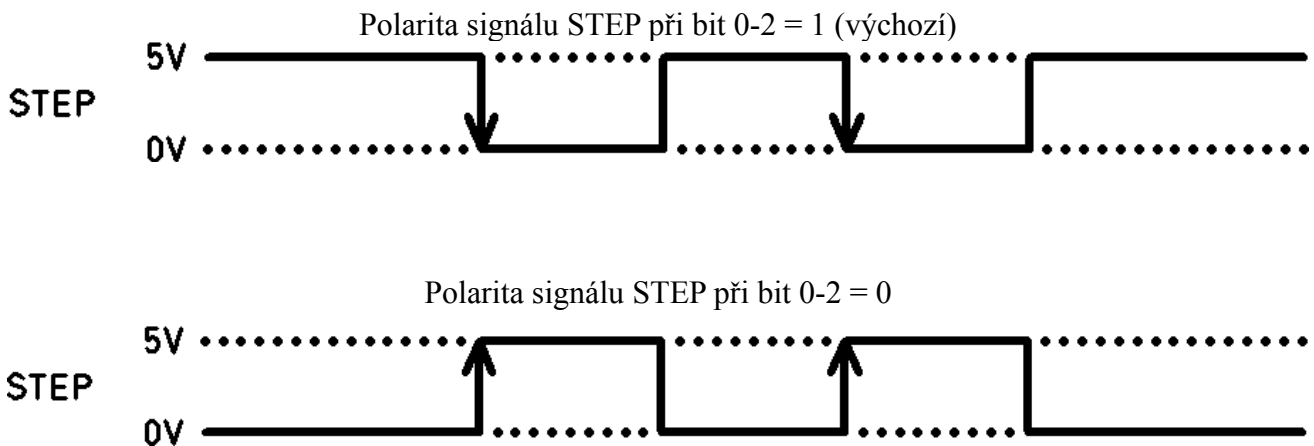
(pokud je výstup nastaven jako brzda, tak je automaticky sepnut po připojení napájení po uplynutí doby nastavené na adrese FD, brzda se používá u strojů s těžším vřeteníkem kde by po vypnutí stroje došlo ke sjetí osy dolů)

### 11.3 Nastavení výstupu signálů STEP/DIR

Nastavení výstupu signálů STEP/DIR na konektorech CNX – CNZ se provádí na adrese 0x05, lze zde měnit polaritu signálů STEP a DIR pro interpolované osy X,Y,Z samostatně pro každou osu.

Změnou polarity signálu DIR se mění směr osy. Hodnoty jsou v hexadecimálním tvaru, výchozí nastavení na FF (všechny bity na 1)

- bit 0 = polarita signálu CLKX
- bit 1 = polarita signálu CLKY
- bit 2 = polarita signálu CLKZ
- bit 3 = nepoužito
- bit 4 = polarita signálu DIRX
- bit 5 = polarita signálu DIRY
- bit 6 = polarita signálu DIRZ
- bit 7 = nepoužito



(šipka značí aktivní hranu, pro drivery s aktivní sestupnou hranou nastavte bit 0-2 na 1 a pro driver s aktivní náběžnou hranou nastavte bit 0-2 na 0, která hrana je pro driver aktivní se dočtete v datasheetu příslušného driveru)

## 12 GVE124 – popis vnitřních instrukcí

(pro tvorbu vlastních uživatelských aplikací)

### 12.1 CPU

procesor ARMv7-M (Cortex-M3) 32bit

### 12.2 Program

IP124 v7/50 2.4.2018 (c) Gravos

### 12.3 Sériový přenos

Sériový přenos 8 bitů, 1 stop bit, bez parity, přenosová rychlost BaudRate a Adresa jednotky jsou fixní.

Přenosovou rychlost lze vybrat při objednání jednotky (viz. Kapitola 10) a adresa jednotky je 0, pro jinou adresu, např. při použití více jednotek najednou, je nutné specifikovat při objednání.

#### 12.3.1 Komunikace

Simplexní, t.j.: nadřazený počítač pošle povel a čeká na odpověď. Až mu dorazí odpověď, tak si ji analyzuje a pošle další, atd...

Nelze posílat příkazy bez čekání na odpověď. Jednotka odpoví vždy co nejdříve, s výjimkou příkazu Halt, kdy odpoví až po zabrzdění.

#### 12.3.2 Zabezpečení přenosu pomocí Checksumu:

Přenos po sériové lince je vhodné zabezpečit, aby v případě nějakého rušení jednotka nebo nadřazený počítač poznali, že se případně přenos příkazu nebo odpovědi nepovedl. Například pokud by z příkazu !0L1000,100 vypadla nějaká nula, pojedle se úplně jinam, což by mohlo mít velmi nepříjemné důsledky.

Pokud je ale aktivován systém kontrolních součtů a nějaké číslo by třeba vypadlo, tak kontrolní součet nebude souhlasit a jednotka příkaz neprovede a nahlásí chybu.

Po zapnutí napájení/resetu jsou kontrolní součty vypnuté.

Zapne se příkazem: !0%+      odpověď je už se součtem: 0,5C

Vypne se příkazem: !0%-,CF      odpověď je už bez součtu: 0

Součet se počítá tak, že se za příkaz dá místo Enteru čárka a sečtou se všechny Ascii hodnoty všech znaků a modulo 256 přidá za čárku součet v hexadecimální podobě, doplní Enterem a odešle.

např: !0A100, = 0x21 + 0x30 + 0x41 + 0x31 + 0x30 + 0x30 + 0x2C = 0x14F, doplníme 4F, výsledek bude !0A100,4F

Ascii kódy lze zjistit např. přímo z příslušenství Windows: Charmap.exe

### 12.3.3 Paketizace příkazů:

Pokud používáte pro přenos dat mezi jednotkou a počítačem převodník USB, je vhodné paketizaci použít. USB porty jsou stavěné trochu jinak než COM porty, které již bohužel pomalu z počítačů mizí. USB porty jsou stavěné sice pro rychlý přenos velkého objemu dat, ale dávkově. Jsou zde časová okna, ve kterých se data přenesou (pokud je zrovna co).

Takže od zadání příkazu do jeho skutečného odeslání vznikne časová "mezera" - je to označované jako Latence, bývá od 1 do 16ms. A při příjmu odpovědi to samé. Takže i přes vysokou přenosovou rychlost USB portů bývá výsledek horší než u Com portu

V případě přenosu malého množství dat třeba pro manipulátory to je většinou nepodstatné, ale pokud budeme chtít např. gravírovat složité křivky, tak jednotka zpracuje příkazy mnohem rychleji než stačíme dodávat data.

Proto je vhodné sdružit více příkazů do jednoho paketu (stringu) a ten poslat najednou. Jednotka odpoví po přijetí konce paketu.

např. včetně kontrolních součtů to může vypadat třeba takto:

!0*S,FA	-start paketu
!0C697,30,0,51	-vektor
!0C692,92,0,54	-vektor
!0C682,151,0,7F	-vektor
!0C665,209,0,84	-vektor
!0C645,268,0,87	-vektor
!0C619,322,0,7F	-vektor
!0C589,375,0,8D	-vektor
!0C553,425,0,80	-vektor
!0C515,471,0,7F	-vektor
!0C471,515,0,7F	-vektor
!0*E10,4D	-konec (bylo 10 příkazů)

a odpověď jednotky: 0,10,E9 - bez chyby, bylo 10 příkazů (a kontrolní součet). Do paketu je možné a účelné dávat jen příkazy typu C (cont.line) a B (brake), ze kterých se vytváří mapa brzdění v koncových bodech vektoru.

Doporučujeme k fréze/manipulátoru, počítač se skutečným(i) sériovým portem, třeba i starší. U stroje mohou ještě dobře posloužit.

V případě použití USB převodníku důrazně doporučujeme, aby byl galvanicky oddělený. USB porty jsou často citlivé na statickou elektřinu.

## 12.4 Reset

Jednotka je po připojení napájení nebo po příkazu J =JUMP na reset cca po 2s ve stavu:

ST0..ST6 = 00	- veškerá přerušení neaktivní
A20	- zrychlení 20000 kr/s2
V1000	- max. rychlost 1000 kr/s
\$512	- short vektory jsou menší než 512 kroků
B50000	- bez omezení rychlosti mezi cont. vektory (pro 50000 hz verzi)
N	- čítač polohy vynulován
00,FF	- výstup vypnutý

## 12.5 Příkazy

! Adresa Příkaz [parametry] Enter  
mezi jednotlivými parametry je čárka

Adr.0 = interpolační jednotka (Adresa může být v rozsahu 0-7, defaultně je adresa 0, při požadavku jiné adresy (např při ovládání více jednotek najednou) je nutné tento požadavek specifikovat při objednání jednotky.

### 12.5.1 Identifikace jednotky

? - **VERSION** dotaz na verzi programu

Q - **QUESTION** dotaz na ID procesoru, vrací řetězec 8 čísel

jejich význam: ddmrrpp

dd = den výroby jednotky

mm = měsíc výroby jednotky

rr = rok výroby jednotky

pp = kolikátá jednotka toho dne

např.: 25020304 znamená 25.2.2003 čtvrtý kus toho dne, toto číslo je jedinečné - neexistují 2 procesory se stejným číslem.

### 12.5.2 Zadávání pohybových vektorů

Lx,y,z - **LINE** vektor (přímka)

x = počet pulsů v ose X v rozsahu -2147483647 až 2147483647

y = počet pulsů v ose Y v rozsahu -2147483647 až 2147483647

z = počet pulsů v ose Z v rozsahu -2147483647 až 2147483647

např.: !0L1000,1000,20

Zvláštní možnost se nabízí při použití vektoru L0,0,0 , který program považuje za normální vektor, i když nemá žádný pohybový efekt. Tento vektor je výhodné zařadit na konec fronty vektorů, kde může indikovat konec zpracování předchozí fronty.

Dokud není přijat, karta hlásí chybu 1, a tudíž fronta před ním není hotová. Jakmile ho karta přijme, ohlásí 0 (OK), a tudíž je fronta před tímto vektorem hotová. Při tomto způsobu je neustále k dispozici bit INTA.

**Cx,y,z** - **CONT.LINE** pokračující vektor (přímka)

x = počet pulsů v ose X v rozsahu -2147483647 az 2147483647

y = počet pulsů v ose Y v rozsahu -2147483647 az 2147483647

z = počet pulsů v ose Z v rozsahu -2147483647 az 2147483647

např.: !0C1000,1000,20

Určit jestli je vektor pokračující je výpočetně složité a časově náročné, a proto to musí určit nadřizený počítač. U pokračujícího vektoru se nesmí příliš změnit úhel, jinak by nebylo fyzikálně možné vektor správně interpretovat.

Jednotka má buffer na 128 CONT. vektorů. ( 1 je vykonáván, a další mohou být ve frontě )

Frontou pokračujících vektorů lze velmi zrychlit práci, protože jednotlivé vektory nemusí neustále zrychlovat z nulové rychlosti a následně opět do nulové rychlosti zpomalovat. Také se tím omezí vibrace stroje a následně se zlepši kvalita obráběného povrchu.

Vektory (L i C) se zadávají v relativních souřadnicích od posledního bodu. (Absolutní souřadnice by představovaly příliš dlouhé řetězce znaků, a proto by klesala skutečná rychlost přenosu informací po sériové lince).

**Tn** - **TIME** prodleva mezi nenavazujícími vektory v milisekundách

n= 1 az 24 milisekund

doporučená hodnota je podle hmotnosti stroje asi 5 az 20 ms

Mezi CONT.vektory tato prodleva není.

např.: !0T5 - prodleva 5ms

Příkaz je modální, platí až do zadání jiné hodnoty.

**\$n** - **SHORT** je hraniční hodnota pro rozlišení krátkého a dlouhého vektoru.

n = 1..2147483647

Dlouhý vektor se snaží dostat pomocí zrychlení A až k maximální rychlosti V.

Krátký vektor se snaží dostat pomocí zrychlení A jen k brzdné rychlosti B na svém konci.

Tímto se stává fronta krátkých vektorů plynulejší, a průjezd libovolnou spojitou křivkou, která je rozumně rozsekána na úsečky je plynulý také.

### 12.5.3 Rychlosti

**An** - **AKCELERATION** zrychlení a zpomalení následujících vektorů

n= tisíců pulsů/s<sup>2</sup>

např.: !0A50 - akcelerace 50000 pulsů/s<sup>2</sup>

Příkaz je modální, platí až do zadání jiné hodnoty.

**Vn** - **VELOCITY** rychlost následujících vektorů

n = 10 až 100000 pulsů/s (záleží na interpolační rychlosti konkrétní jednotky)

např.: !0V10000 - rychlost 10000 pulsů/s

Příkaz je modální, platí až do zadání jiné hodnoty.

**VLx,y,z,a** – **VELOCITY LIMIT** rychlostní limit

Omezení max. rychlosti, které můžou jednotlivé osy dosáhnout

X = max. rychlost v tis. pulsů/s osy X v rozsahu 10 - 100000

Y = max. rychlost v tis. pulsů/s osy Y v rozsahu 10 - 100000

Z = max. rychlost v tis. pulsů/s osy Z v rozsahu 10 - 100000

např. !0VL25000,25000,30000

**Bn** - **BRAKE** rychlost, na kterou má vektor dobrzdit, pokud za ním ve frontě je další Cont.vektor.

Pokud za ním není další, tak stejně dobrzdí do nuly.

n = 10 až 100000 pulsů/s

To má význam hlavně u navazujících vektorů, kdy je nutné před zatáčkou přibrzdit, ale ne úplně. Příkaz je modální, platí až do zadání jiné hodnoty.

### 12.5.4 Korekce rychlostí

**VK** - **VELOCITY CORECTION** korekce rychlosti podle směru

rychlosti jsou určeny podle  $d = \sqrt{dx*dx + dy*dy + dz*dz} / d$

Použití hlavně pro pohyby za předpokladu že všechny osy jsou lineární

**VN** - **NO VELOCITY CORECTION** rychlosti bez korekcí podle směru, kde rychlost pro každý vektor musí určit nadřazené PC s ohledem na směr pohybů jednotlivých os.

### 12.5.5 Změny rychlosti během pohybu

**XA** - **EXCHANGE** změna rychlosti u všech vektorů ve frontě na poslední zadanou rychlost V.

**XU** - **EXCHANGE UP** změna rychlosti u všech vektorů ve frontě. Hodí se pro změnu parametrů za chodu. Rychlost se zvětší o 1/16 (6,25%) současného stavu

**XD** - **EXCHANGE DN** změna rychlosti u všech vektorů ve frontě. Hodí se pro změnu parametrů za chodu. Rychlost se zmenší o 1/16 (6,25%) současného stavu

**XMn** - **EXCHANGE MULTIPLIER** násobitel rychlosti, parametr n je násobitel rychlosti v procentech.

např. !0XM100 nastaví rychlost na poslední zadanou rychlost příkazem V

např. !0XM200 nastaví rychlost na dvojnásobek (200%) rychlosti nastavené příkazem V

Pokud je nastaven rychlostní limit příkazem VL, rychlosti jednotlivých os můžou dosáhnout max. rychlosti nastavené tímto limitem.

## 12.5.6 Poloha

**P** - **POSITION** dotaz na polohu X,Y. Odpovědí je okamžitá absolutní poloha x,y , takže během chodu nějakého vektoru se neustále mění. Po zastavení je hodnota stabilní. Vhodné pro kreslení okamžité pozice nástroje v rovině XY.

**PF** - **POSITION** dotaz na polohu X,Y,Z. Odpovědí je okamžitá absolutní poloha x,y,z , takže během chodu nějakého vektoru se neustále mění. Po zastavení je hodnota stabilní.

**N** - **NULL ALL** vynuluje všechny 3 osy čítače pozice. Nelze použít za chodu vektoru (při RUN=1)

**Nan** - **NULL AXIS** Vynuluje nebo nastaví čítač pozice vybrané osy. Parametr a je osa, které se má čítač nastavit (X,Y,Z)

parametr n je v rozsahu -2147483647 až 2147483647

např. !0NX0 – vynuluje čítač pozice osy X

např. !0NX100 – nastaví čítač pozice na hodnotu 100

## 12.5.7 Nalezení spínače osy - referenční pohyb

**Wn** - **Switch** nalezení spínače osy (referenční pohyb). Jednotka odpoví až po ukončení reference

Příkaz je ve formátu Wn1,n2,n3,n4,n5

parametry:

n1 = Osa (x,y,z)

n2 = Max. délka a směr kterou osa jede ke spínači [pulsy]

n3 = Rychlost ke spínači [pulsů/s]

n4 = Max. Délka kterou osa jede od spínače [pulsy]

n5 = Rychlost od spínače [pulsů/s]

např.reference osy Y: !0WY,-20000,1000,2000,500



## 12.5.8 Obsluha fronty a zpracování vektorů

**K** - **KEEP** {obdoba PUSH} Zachytí v operační paměti stav fronty vektorů po přerušení a ST4,ST5 a ST6. Potom smaže ST4=00,ST5=00 a ST6=00.

Vyhradí v operační paměti místo pro 1 vektor, takže je možno opět zadávat vektory, ale již jen typu L (C ne). KEEP lze použít bez odpovídajícího RESTORE jen jednou. Nelze použít za chodu.

**R** - **RESTORE** {obdoba POP} Inverzní rutina ke KEEP. Obnoví stav operační paměti s frontou vektorů a ST4,ST5,ST6 tak, jak byla uložena příkazem KEEP. Nelze použít za chodu.

Tato dvojice inverzních rutin umožňuje transparentci vektorů po přerušení.

Např.: Obsluha zastaví obrábění tlačítkem STOP nebo příkazem HALT apod. Potom je obrábění zastaveno, ale v jednotce je ještě zbytek vektorů ve frontě. Tento zbytek lze dokončit příkazem GO, nebo smazat příkazem DELETE, ale někdy je potřeba zvednout nástroj a nezničit zbytek fronty. Potom je potřeba zachytit stav paměti, vymazat ji, udělat zadané vektory (vzhůru a zpět dolů) a potom obnovit paměť a pokračovat v obrábění.

např.:

!0H	zastaví vektor	
!0P	zjistí souřadnice zastavení	(kde to jsme?)
!0SR4	zjištění stavu systému přerušení	(a proč se to stalo?)
!0SR5	zjištění stavu systému přerušení	
!0SR6	zjištění stavu systému	
!0SW0,1A	nová maska přerušení	
!0SW1,3F	nová maska přerušení	
!0K	zachytí stav operační paměti	
!0L0,0,-1000	zvedne nástroj (pro jeho výměnu)	
....tady se čeká na reakci uživatele....		
....a když se rozhodne pokračovat třeba změněnou rychlostí....		
!0L0,0,1000	spustí nástroj	
!0R	obnoví operační paměť	
!0V500	nastaví novou rychlost budoucích vektorů	
!0XA	nastaví tuto rychlost i pro zbytek vektorů ve frontě	
!0SW0,A2	normální maska přerušení	
!0SW1,78	normální maska přerušení	
!0G	pokračování už jinou rychlostí	

**H** - **HALT** zastavení zpracovávaného vektoru, pokud nějaký běží Nastaví bit INTA=1. Bit RUN signalizuje, zda byl příkaz HALT použit za chodu (1), nebo ne (0). Odpoví až po zastavení. To může trvat i dost dlouho - neztracovat zatím komunikaci.

Příkazem HALT se zároveň nastaví bit INTRCOM pro účel identifikace přerušení.

**D** - **DELETE** smaže veškeré vektory ve frontě. Určeno pro smazání zbytku fronty po přerušení. Smaže všechny příznaky přerušení (ST4=00(hex), ST5=00(hex), ST6=00(hex)) Čítač pozice neovlivní. Nelze použít za chodu.

**G** - **GO** nastartuje dokončení zastaveného vektoru a zbytku fronty jen pokud INTA=1, jinak bez efektu. Smaže bit INTA=0,INTRCOM=0,ST4=00,ST5=00.

### 12.5.9 Obsluha paměti EEPROM

**ERn** - **READ BYTE** přečte byte z EEPROM na adrese n a pošle jej po sériové lince

n = 00..FF(hex)

**EWn,x** - **WRITE BYTE** zapíše byte x EEPROM na adresu n

n = 00..FF(hex) x = 00..FF(hex)

### 12.5.10 Ovládání relé

**00,n** - **OUTPUT** zapíše byte x v hexadecimálním tvaru na výstupní port 0 jsou aktivní v log.0:

bit 0 = CNOOUT - OUT0 (v kontrolérech Gravos včetně)

bit 1 = CNOOUT - OUT3 (v kontrolérech Gravos brzda)

bit 2 = CNOOUT - OUT1 (v kontrolérech Gravos chlazení)

bit 3 = CNOOUT - OUT2 (v kontrolérech Gravos ofuk)

bit 4 – 7 = nepoužit

výstupy jsou aktivní v log.0, po zapnutí jsou neaktivní log.1

např. spuštění chlazení (OUT1): !000,FB

vypnutí všeho: !000,FF

### 12.5.11 Čtení stavu vstupů a obsluha přerušení

**I1** - **INPUT** přečte vstupní port 1, odpovědí je stav portu v hexadecimálním tvaru

bit 0 = CNIN - IN0 Intr0 (v kontrolérech Gravos RefX)

bit 1 = CNIN - IN1 Intr1 (v kontrolérech Gravos RefY)

bit 2 = CNIN - IN2 Intr2 (v kontrolérech Gravos RefZ)

bit 3 -7 není použit

Intry nedělají nic jiného, než že při své aktivaci přinutí interpolátor zabrzdit (po rampě). Je zde popsáno, jak využívá Intry systém Gravos, to by však nemělo být omezující, lze je použít libovolně jinak. Toto info je jen pro případnou snahu o kompatibilitu.

**SRn** - **STATUS READ** přečte status n = 0..5. odpovědí je hodnota zadaného status slova

**SW<sub>n,x</sub> - STATUS WRITE** zapíše do statusu  $n = 0..5$ , byte  $x$  (v hex.tvaru)

Status slova:

ST0 = povolení uživatel. přerušení INTR0-2 ( 0 = zakázáno )

ST2 = polarita uživatel. přerušení INTR0-2 ( 0 = aktivní v log.0 )

ST4 = příčina přerušení INTR0-2 ( 0 = přerušení nebylo )

Jednotlivá přerušení korespondují se vstupy. Pomocí přečtení vstupů lze přečíst okamžitý stav. Každé aktivované přerušení zastaví pohyb a nastaví bit INTA, aby o tom řídicí SW věděl.

Libovolný Intr není nutné použít, (lze zamaskovat) a je ho možno použít jako obecný vstupní bit.

**SR6 - STATUS READ** přečte ST6 význam jednotlivých bitů: (ostatní jsou nepoužité)

STOP = 0 žádost o zastavení

FREE = 1 příznak volného str.času

INTCOM = 4 nastavuje se po přerušení HALTem

RUN = 6 je zpracováván vektor

INTA = 7 akceptováno zastavení

pro uživatele mají význam především bity RUN a INTA

INTA=0 RUN=0 ;nic není spuštěno, klidový stav

INTA=0 RUN=1 ;provozní stav, jsou zpracovávány vektory

INTA=1 RUN=0 ;bylo přerušeno, při brzdění vektory doběhly

INTA=1 RUN=1 ;bylo přerušeno, zbytek vektorů je ve frontě

**SW6 - STATUS WRITE** zapíše do ST6, byte  $x$  (v hex.tvaru) raději nepoužívat, lépe použít instrukce G,D,H apod...

**F - FLAG** to samé jako SR6, ale je doplněn stav fronty vektorů - bit 5

log.1 = fronta je plná - nelze přijmout vektor

log.0 = do fronty se další vektor vejde

## 12.5.12 Příkazy pro opravu chyb komunikace

**@ - INDEX** pošle index posledního příkazu. Všechny příkazy jsou indexovány modulo 256.

V případě nejistoty, zda příkaz do Interpolátoru dorazil, je možné vyžádat tento index a porovnat s vlastním indexováním v programu, a tak zjistit, zda ho interpolátor přijal nebo ne. Většina příkazů se dá zopakovat (A,V,PF atd..), ale zadávání polohy ne, to se musí v případě chyby přenosu exaktně dohledat, jinak by se jelo jinam.

**> - REPEAT** - zopakuje poslední přijatý příkaz a odpověď na něj. Toto se hodí, pokud dojde k chybě přenosu a nadřizované počítači přijde místo odpovědi nějaký nesmysl.

**J - JUMP** na RESET zresetuje včetně vynulování čítače polohy.

## 12.6 Odpovědi

1 hexadecimální znak 0..F [další vyžadované parametry] Enter (0Dh) mezi jednotlivými parametry je čárka. Odpověď je odeslána ihned po zadání příkazu.

Jedině pro příkaz HALT je odpověď odeslána až po vykonání instrukce.

INTA, 3 bitový kód chyby

bit 3, 2 .. 0

kód chyby:

0 = OK (žádná chyba)

1 = fronta je plná, nelze zařadit další vektor je nutné počkat, zopakovat

2 = příkaz nepřišel celý včas, přetržení komunikace (zafunguje WATCH DOG)

3 = neznámý příkaz

4 = chyba syntaxe

5 = parametr mimo meze

6 = pro Go, není co spustit

7 = za chodu vektoru nelze

Např.:

Příkaz	Odpověď	Pozn.
!0L1000,0,0	0	OK
!0PF	0,-1000,2000,50	OK
!0C100,20,0	1	vektor nebyl přijat (je nutné ho opakovat)
!0V1000	8	OK, ale je přerušeno (INTA=1)
!0SW8,F1	5	parametr mimo meze
!0SR2	0,2B	OK

# Obsah

1	SPECIFIKACE.....	1
2	APLIKACE.....	1
3	SOUČÁST DODÁVKY.....	2
4	ROZMĚRY.....	2
5	PŘEHLED.....	2
6	POPIS KONEKTORŮ.....	3
7	POPIS VÝVODŮ.....	3
8	FUNKCE LED SIGNALIZACE.....	4
9	PŘÍKLADY DOPORUČENÉHO ZAPOJENÍ.....	5
9.1	Připojení pohonů zařízení (CNX – CNZ).....	5
9.1.1	Připojení krokových motorů.....	5
9.1.2	Časování signálů KROK a SMĚR.....	5
9.2	Zapojení vstupů (CNIN).....	6
9.2.1	Připojení referenčních spínačů.....	6
9.3	Zapojení výstupu (CNOU).....	6
9.3.1	Připojení signálu START k frekvenčnímu měniči.....	6
	Připojení signálu START a 0-10V k frekvenčnímu měniči.....	6
9.3.3	Připojení stykače pro spínání větší zátěže než 30VDC/2A.....	7
9.3.4	Připojení elmag. ventilu.....	7
9.4	Kabel k připojení GVE124 k PC (CN1).....	7
9.5	Napájení (CNSUP).....	8
10	MOŽNÉ VERZE JEDNOTKY.....	8
11	NASTAVENÍ FUNKCE VÝSTUPŮ.....	9
11.1	Adresy EEPROM pro výstupy.....	9
11.2	Hodnoty nastavení pro RELÉ.....	9
11.3	Nastavení výstupu signálů STEP/DIR.....	10
12	GVE124 – POPIS VNITŘNÍCH INSTRUKCÍ.....	11
12.1	CPU.....	11
12.2	Program.....	11
12.3	Sériový přenos.....	11
12.3.1	Komunikace.....	11
12.3.2	Zabezpečení přenosu pomocí Checksumu.....	11
12.3.3	Paketizace příkazů.....	12
12.4	Reset.....	13
12.5	Příkazy.....	13
12.5.1	Identifikace jednotky.....	13
12.5.2	Zadávání pohybových vektorů.....	13
12.5.3	Rychlosti.....	15
12.5.4	Korekce rychlostí.....	15
12.5.5	Změny rychlosti během pohybu.....	15
12.5.6	Poloha.....	16
12.5.7	Nalezení spínače osy - referenční pohyb.....	16
12.5.8	Obsluha fronty a zpracování vektorů.....	17
12.5.9	Obsluha paměti EEPROM.....	18
12.5.10	Ovládání relé.....	18
12.5.11	Čtení stavu vstupů a obsluha přerušení.....	18
12.5.12	Příkazy pro opravu chyb komunikace.....	19
12.6	Odpovědi.....	20

© Gravos CNC s.r.o

Poslední změna 18.5.2018 [WWW.GRAVOS.CZ](http://WWW.GRAVOS.CZ)