



# LinuxCNC

## Spindel Konfiguration

Meine verwendeten Komponenten.

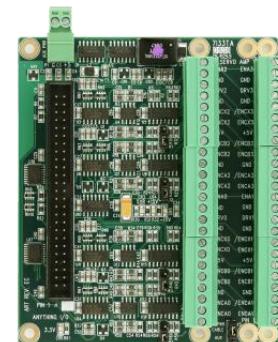
Servodrive

Estun EDB-10A + Estun Servomotor 1KW 1000 U/min



Mesa Karten

5i20 + 7i33



# Theorie - Konfiguration der Spindel

Ich konfiguriere in der HAL eine Spindel die das PWM an der Regler gibt.

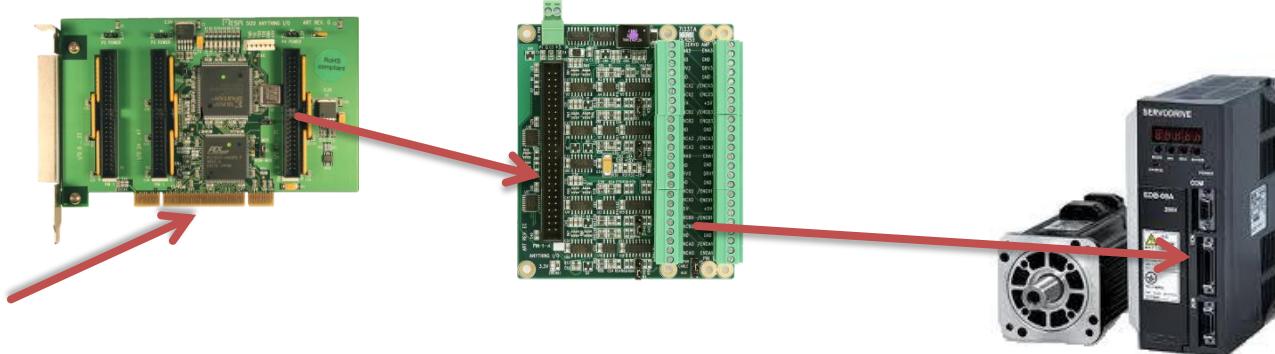
Max Drehzahl = 1500 U/min somit wären +10V = 1500 U/min Rechtslauf und -10V = 1500 U/min Linkslauf.

Des Weiteren will ich das LinuxCNC erst mit den Achsbewegungen beginnt wenn auch die eingestellte Drehzahl erreicht ist.

Außerdem möchte ich auch noch eine Rampen haben mit der ich die Beschleunigung und das Bremsen der Spindel einstellen kann.

Aber wie





## Kommunikation

Die 5i20 steckt im PCI Steckplatz vom PC und kommuniziert über Hostmot2 mit LinuxCNC.

Beim Start von LinuxCNC wird über eine Zeile in der HAL das jeweilige Bitfile in die Karte geladen.

Das Bitfile entscheidet darüber welche Funktionen an den einzelnen Pins zur Verfügung stehen.

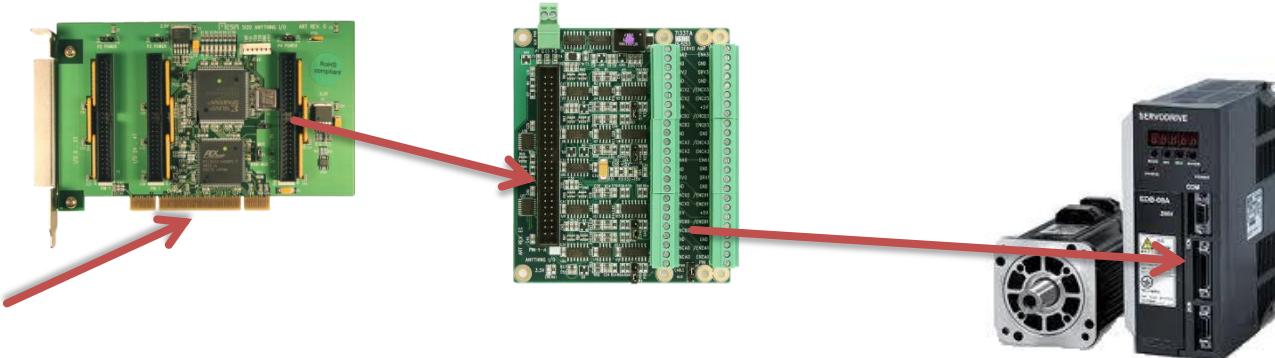
Hier verwende ich folgende Zeile zur Initialisierung der Karte.

```
loadrt hm2_pci config="firmware=hm2/5i20/SVST8_4.BIT num_encoders=4 num_pwmgens=4 num_stepgens=0,,
```

PIN	IO#	Module	Chan Func
P2-1	0	Encoder	1 Phase B (in)
P2-3	1	Encoder	1 Phase A (in)
P2-5	2	Encoder	0 Phase B (in)
P2-7	3	Encoder	0 Phase A (in)
P2-9	4	Encoder	1 Index (in)
P2-11	5	Encoder	0 Index (in)
P2-13	6	PWMGen	1 PWM/Up (out)
P2-15	7	PWMGen	0 PWM/Up (out)
P2-17	8	PWMGen	1 Dir/Down (out)
P2-19	9	PWMGen	0 Dir/Down (out)
P2-21	10	PWMGen	1 Enable (out)
P2-23	11	PWMGen	0 Enable (out)
P2-25	12	Encoder	3 Phase B (in)
P2-27	13	Encoder	3 Phase A (in)
P2-29	14	Encoder	2 Phase B (in)
P2-31	15	Encoder	2 Phase A (in)
P2-33	16	Encoder	3 Index (in)
P2-35	17	Encoder	2 Index (in)
P2-37	18	PWMGen	3 PWM/Up (out)
P2-39	19	PWMGen	2 PWM/Up (out)
P2-41	20	PWMGen	3 Dir/Down (out)
P2-43	21	PWMGen	2 Dir/Down (out)
P2-45	22	PWMGen	3 Enable (out)
P2-47	23	PWMGen	2 Enable (out)

PIN	IO#	Module	Chan Func
P3-1	24	Encoder	5 Phase B (in)
P3-3	25	Encoder	5 Phase A (in)
P3-5	26	Encoder	4 Phase B (in)
P3-7	27	Encoder	4 Phase A (in)
P3-9	28	Encoder	5 Index (in)
P3-11	29	Encoder	4 Index (in)
P3-13	30	PWMGen	5 PWM/Up (out)
P3-15	31	PWMGen	4 PWM/Up (out)
P3-17	32	PWMGen	5 Dir/Down (out)
P3-19	33	PWMGen	4 Dir/Down (out)
P3-21	34	PWMGen	5 Enable (out)
P3-23	35	PWMGen	4 Enable (out)
P3-25	36	Encoder	7 Phase B (in)
P3-27	37	Encoder	7 Phase A (in)
P3-29	38	Encoder	6 Phase B (in)
P3-31	39	Encoder	6 Phase A (in)
P3-33	40	Encoder	7 Index (in)
P3-35	41	Encoder	6 Index (in)
P3-37	42	PWMGen	7 PWM/Up (out)
P3-39	43	PWMGen	6 PWM/Up (out)
P3-41	44	PWMGen	7 Dir/Down (out)
P3-43	45	PWMGen	6 Dir/Down (out)
P3-45	46	PWMGen	7 Enable (out)
P3-47	47	PWMGen	6 Enable (out)

PIN	IO#	Module	Chan Func
P4-1	48	StepGen	0 Step (out)
P4-3	49	StepGen	0 Dir (out)
P4-5	50	StepGen	0 StepTable 2 (out)
P4-7	51	StepGen	0 StepTable 3 (out)
P4-9	52	StepGen	0 StepTable 4 (out)
P4-11	53	StepGen	0 StepTable 5 (out)
P4-13	54	StepGen	1 Step (out)
P4-15	55	StepGen	1 Dir (out)
P4-17	56	StepGen	1 StepTable 2 (out)
P4-19	57	StepGen	1 StepTable 3 (out)
P4-21	58	StepGen	1 StepTable 4 (out)
P4-23	59	StepGen	1 StepTable 5 (out)
P4-25	60	StepGen	2 Step (out)
P4-27	61	StepGen	2 Dir (out)
P4-29	62	StepGen	2 StepTable 2 (out)
P4-31	63	StepGen	2 StepTable 3 (out)
P4-33	64	StepGen	2 StepTable 4 (out)
P4-35	65	StepGen	2 StepTable 5 (out)
P4-37	66	StepGen	3 Step (out)
P4-39	67	StepGen	3 Dir (out)
P4-41	68	StepGen	3 StepTable 2 (out)
P4-43	69	StepGen	3 StepTable 3 (out)
P4-45	70	StepGen	3 StepTable 4 (out)
P4-47	71	StepGen	3 StepTable 5 (out)



## Kommunikation

Die 5i20 steckt im PCI Steckplatz vom PC und kommuniziert über Hostmot2 mit LinuxCNC.

Beim Start von LinuxCNC wird über eine Zeile in der HAL das jeweilige Bitfile in die Karte geladen.

Das Bitfile entscheidet darüber welche Funktionen an den einzelnen Pins zur Verfügung stehen.

Hier verwende ich folgende Zeile zur Initialisierung der Karte.

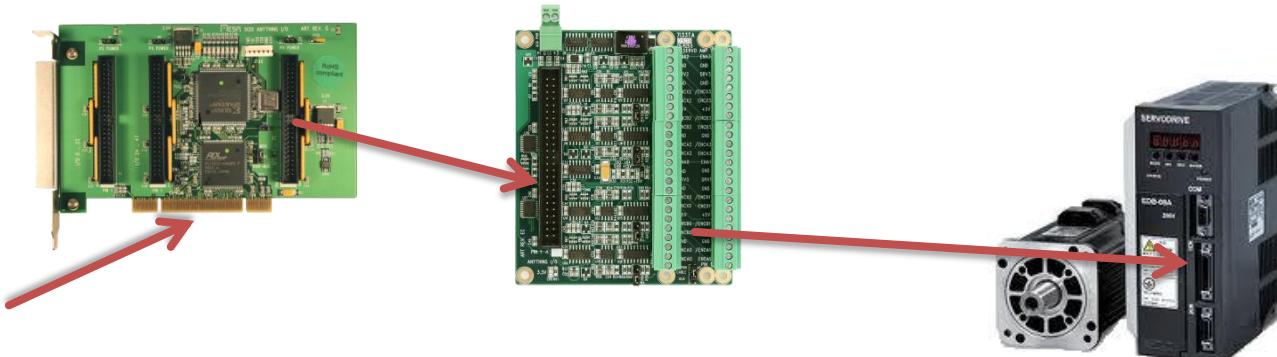
```
loadrt hm2_pci config="firmware=hm2/5i20/SVST8_4.BIT num_encoders=4 num_pwmgens=4 num_stepgens=0,,
```

```
loadrt trivkins
```

```
loadrt [EMCMOT]EMCMOT servo_period_nsec=[EMCMOT]SERVO_PERIOD num_joints=[TRAJ]AXES  
loadrt hostmot2
```

```
loadrt hm2_pci config="firmware=hm2/5i20/SVST8_4.BIT num_encoders=4 num_pwmgens=4 num_stepgens=0"
```

```
setp hm2_5i20.0.pwmgen.pwm_frequency 20000  
setp hm2_5i20.0.pwmgen.pdm_frequency 6000000  
setp hm2_5i20.0.watchdog.timeout_ns 5000000  
addf hm2_5i20.0.read servo-thread  
addf hm2_5i20.0.write servo-thread  
addf motion-command-handler servo-thread  
addf motion-controller servo-thread
```



## Kommunikation

Die 5i20 steckt im PCI Steckplatz vom PC und kommuniziert über Hostmot2 mit LinuxCNC.

Beim Start von LinuxCNC wird über eine Zeile in der HAL das jeweilige Bitfile in die Karte geladen.

Das Bitfile entscheidet darüber welche Funktionen an den einzelnen Pins zur Verfügung stehen.

Hier verwende ich folgende Zeile zur Initialisierung der Karte.

```
loadrt hm2_pci config="firmware=hm2/5i20/SVST8_4.BIT num_encoders=4 num_pwmgens=4 num_stepgens=0,,
```

Somit kann ich die 7i33 mit einem 50 pol Flachbandkabel an den P2 anschließen. P3 und P4 nutze ich hier erstmal nicht.

Am P2 der 5i20 stehen wie rechts abgebildet nun Folgende Funktionen zur Verfügung. 4 x PWM Ausgang / 4 x Dir Ausgang 4 x Enable Ausgang  
4 x Encoder Eingang

Die Pin Belegung ist kompatibel mit der 7i33 es muss also nichts im Bitfile angepasst werden.

Das PWM und DIR wird von der 7i33 in ein Analogsignal

$\pm 10V$  umgewandelt. Somit lässt sich mein Regler über Speed ansteuern. Ich habe DRVO von der 7i33 mit dem Analogeingang (1CN) des Reglers verbunden.

Damit LinuxCNC nun auch den Encoder vom Motor sieht habe ich die Encoder Signale A/B/Z vom Regler Anschluss 1CN an die 7i33 (ENC0A / ENC0B und ENC0X )angeschlossen.

In der HAL muss ich dann die PINS vom hm2\_5i20.0.pwmgen.00. und hm2\_5i20.0.encoder.00. verwenden.

# HAL - Konfiguration der Spindel

Als erstes erstelle ich in der INI eine Sektor [SPINDLE\_9] und einige Variablen die dann von der HAL gelesen werden.  
Damit kann man später die Spindel einfacher parametrieren.

```
[SPINDLE_9]
ENCODER_SCALE    = 166.666667    (Mein Encoder liefert 10000 Impulse pro Umdrehung . Gerechnet habe ich 10000 Impulse / 60 Sekunden = 166,66~ Impulse/Sekunde)
OUTPUT_SCALE     = 1500          (bei ±10V die maximale Drehzahl)
BESCHLEUNIGUNG   = 1000          (mit welcher Beschleunigung und Bremswirkung soll gefahren werden)
MAX_FEHLER       = 20            (akzeptierter Fehler zwischen Sollwert und Istwert)
```

In der HAL erstelle ich mit den Komponenten limit2 und near die Rampenfunktion und die Drehzahlprüfung.

Dann verknüpfe ich das mit den Linux Motion und Mesa PINS. Auch die Parameter der einzelnen Komponenten setze ich in der HAL bzw. lese die Variablen aus der INI.

```
# SPINDLE
setp hm2_5i20.0.encoder.00.counter-mode 0
setp hm2_5i20.0.encoder.00.filter 1
setp hm2_5i20.0.encoder.00.index-invert 0
setp hm2_5i20.0.encoder.00.index-mask 0
setp hm2_5i20.0.encoder.00.index-mask-invert 0
setp hm2_5i20.0.encoder.00.scale [SPINDLE_9]ENCODER_SCALE           (hier wird z.B. der Wert aus der INI übernommen)
setp hm2_5i20.0.pwmgen.00.output-type 1
setp hm2_5i20.0.pwmgen.00.scale [SPINDLE_9]OUTPUT_SCALE             (hier wird z.B. der Wert aus der INI übernommen)

loadrt limit2 names=spindle-ramp
loadrt near names=spindle-at-speed
addf spindle-ramp servo-thread
addf spindle-at-speed servo-thread
setp spindle-ramp.maxv [SPINDLE_9]BESCHLEUNIGUNG
setp spindle-at-speed.difference [SPINDLE_9]MAX_FEHLER               (limit2 laden und mit Name versehen)
                                         (near laden und mit Name versehen)
                                         (Funktion in den Servo-Thread laden damit diese überhaupt funktioniert)
                                         (Funktion in den Servo-Thread laden damit diese überhaupt funktioniert)
                                         (hier wird z.B. der Wert aus der INI übernommen)
                                         (hier wird z.B. der Wert aus der INI übernommen)

net sollwert      spindle-at-speed.in1 <= motion.spindle-speed-out => spindle-ramp.in      (Sollwert in das near und limit2 geben damit diese rechnen können)
net drehzahl-ok   spindle-at-speed.out => motion.spindle-at-speed
net pwm-ausgang   spindle-ramp.out => hm2_5i20.0.pwmgen.00.value                      (BIT vom near in Motion geben damit Linux erst los fährt wenn Drehzahl OK ist)
net encoder-fb    motion.spindle-speed-in <= hm2_5i20.0.encoder.00.velocity => spindle-at-speed.in2 (limit2 Ergebnisse an den PWM Ausgang geben)
                                         (Encodergeschwindigkeit in Motion und near geben.

net spindle-on    motion.motion-enabled => hm2_5i20.0.pwmgen.00.enable                  (Wenn Maschine an dann ist der PWM Ausgang auch aktiv)

net estop-out     <= iocontrol.0.user-enable-out
net estop-out     => iocontrol.0.emc-enable-in
```

# HAL - Konfiguration der Spindel

```
#21.12.2017 Spindel Konfiguration
#http://talla83.de/linuxcnc/config/config.htm
#Mesa 5i20 + 7i33

loadrt trivkins
loadrt [EMCMOT]EMCMOT servo_period_nsec=[EMCMOT]SERVO_PERIOD num_joints=[TRAJ]AXES
loadrt hostmot2
loadrt hm2_pci config="firmware=hm2/5i20/SVST8_4.BIT num_encoders=4 num_pwmgens=4 num_stepgens=0"

setp hm2_5i20.0.pwmgen.pwm_frequency 20000
setp hm2_5i20.0.pwmgen.pdm_frequency 6000000
setp hm2_5i20.0.watchdog.timeout_ns 5000000

addir hm2_5i20.0.read servo-thread
addir hm2_5i20.0.write servo-thread
addir motion-command-handler servo-thread
addir motion-controller servo-thread

# SPINDLE

setp hm2_5i20.0.encoder.00.counter-mode 0
setp hm2_5i20.0.encoder.00.filter 1
setp hm2_5i20.0.encoder.00.index-invert 0
setp hm2_5i20.0.encoder.00.index-mask 0
setp hm2_5i20.0.encoder.00.index-mask-invert 0
setp hm2_5i20.0.encoder.00.scale [SPINDLE_9]ENCODER_SCALE
setp hm2_5i20.0.pwmgen.00.output-type 1
setp hm2_5i20.0.pwmgen.00.scale [SPINDLE_9]OUTPUT_SCALE

loadrt limit2 names=spindle-ramp
loadrt near names=spindle-at-speed
addir spindle-ramp servo-thread
addir spindle-at-speed servo-thread
setp spindle-ramp.maxv [SPINDLE_9]BESCHLEUNIGUNG
setp spindle-at-speed.difference [SPINDLE_9]MAX_FEHLER

net sollwert spindle-at-speed.in1 <= motion.spindle-speed-out => spindle-ramp.in
net drehzahl-ok spindle-at-speed.out => motion.spindle-at-speed
net pwm-ausgang spindle-ramp.out => hm2_5i20.0.pwmgen.00.value
net encoder-fb motion.spindle-speed-in <= hm2_5i20.0.encoder.00.velocity => spindle-at-speed.in2

net spindle-on motion.motion-enabled => hm2_5i20.0.pwmgen.00.enable

net estop-out <= iocontrol.0.user-enable-out
net estop-out => iocontrol.0.emc-enable-in
```

# INI - Konfiguration der Spindel

```
#21.12.2017 Spindel Konfiguration
#http://talla83.de/linuxcnc/config/config.htm
#Mesa 5i20 + 7i33

[EMC]
MACHINE = Spindel
DEBUG = 0

[DISPLAY]
DISPLAY = axis
POSITION_OFFSET = RELATIVE
POSITION_FEEDBACK = ACTUAL
MAX_FEED_OVERRIDE = 2.000000
MAX_SPINDLE_OVERRIDE = 1.000000
MIN_SPINDLE_OVERRIDE = 0.500000
INTRO_GRAPHIC = linuxcnc.gif
INTRO_TIME = 5
PROGRAM_PREFIX = /home/cnc/linuxcnc/nc_files
INCREMENTS = 5mm 1mm .5mm .1mm .05mm .01mm .005mm
POSITION_FEEDBACK = ACTUAL
DEFAULT_LINEAR_VELOCITY = 6.000000
MAX_LINEAR_VELOCITY = 25.000000
MIN_LINEAR_VELOCITY = 0.500000
DEFAULT_ANGULAR_VELOCITY = 12.000000
MAX_ANGULAR_VELOCITY = 180.000000
MIN_ANGULAR_VELOCITY = 1.666667
EDITOR = gedit
GEOMETRY = xyz

[FILTER]
PROGRAM_EXTENSION = .png,.gif,.jpg Greyscale Depth Image
PROGRAM_EXTENSION = .py Python Script
png = image-to-gcode
gif = image-to-gcode
jpg = image-to-gcode
py = python

[TASK]
TASK = milltask
CYCLE_TIME = 0.010

[RS274NGC]
PARAMETER_FILE = linuxcnc.var

[EMCMOT]
EMCMOT = motmod
COMM_TIMEOUT = 1.0
COMM_WAIT = 0.010
SERVO_PERIOD = 1000000
```

```
[HOSTMOT2]
# **** This is for info only ****
# DRIVER0=hm2_pci
# BOARD0=5i20

[HAL]
HALUI = halui
HALFILE = Spindel.hal
HALFILE = custom.hal
POSTGUI_HALFILE = postgui_call_list.hal
SHUTDOWN = shutdown.hal

[HALUI]

[TRAJ]
AXES = 3
COORDINATES = X Y Z
LINEAR_UNITS = mm
ANGULAR_UNITS = degree
CYCLE_TIME = 0.010
DEFAULT_VELOCITY = 2.50
MAX_LINEAR_VELOCITY = 25.00

[EMCIO]
EMCIO = io
CYCLE_TIME = 0.100
TOOL_TABLE = tool.tbl

[SPINDLE_9]
ENCODER_SCALE      = 166.6666667
OUTPUT_SCALE       = 1500
BESCHLEUNIGUNG    = 1000
MAX_FEHLER        = 20

[AXIS_0]
HOME_SEQUENCE = 0
# Axis X
[AXIS_1]
HOME_SEQUENCE = 0
# Axis Y
[AXIS_2]
HOME_SEQUENCE = 0
# Axis Z
```



# LinuxCNC

## Spindel Konfiguration

<http://talla83.de/linuxcnc/config/config.htm>

<http://linuxcnc.org/docs/2.7/html/man/man9/hostmot2.9.html>

<http://linuxcnc.org/docs/2.7/html/man/man9/near.9.html>

<http://linuxcnc.org/docs/2.7/html/man/man9/limit2.9.html>

<http://linuxcnc.org/docs/2.7/html/man/man9/motion.9.html>