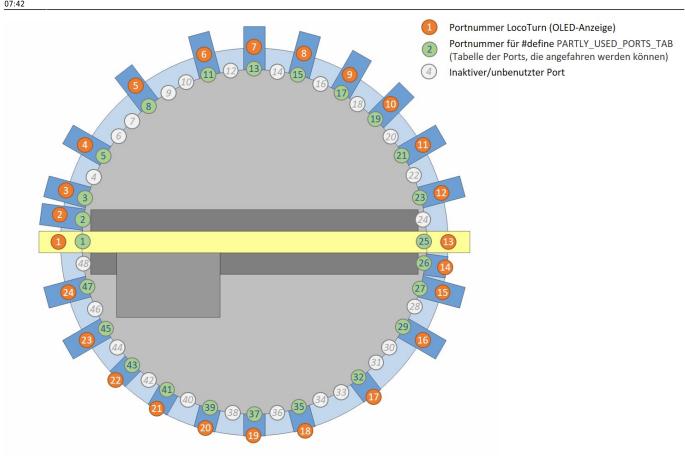
Parameter zur Einstellung der Drehscheibe

Aus der Vielzahl der Konfigurationsvariablen (#defines) hier ein paar wesentliche:

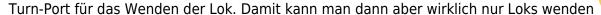
- Einige #defines wie USE_DCC, USE_SOUND, USE_SERIAL_INPUT etc. steuern, was der Sketch alles später an Funktionen bietet, also, ob DCC, Sound und Eingaben über den seriellen Monitor verwendet werden.
- Das #define USE_POTI muss auf 0 gesetzt werden, wenn kein Poti angeschlossen ist, sonst kann es sein, dass die Drehscheibe nach dem ersten Kalibrieren und Anfahren des ersten Ports "unkontrolliert herumschleicht".
- Zentrale Parameter sind die Anzahl der benötigten Ports #define PORT_CNT bzw. #define PARTLY_USED_PORTS (falls nicht alle angefahren werden sollen) und #define PARTLY_USED_PORTS_TAB (Tabelle der Ports, die angefahren werden können).
- Über das #define **DCC_OFFSET** kann ein Offset eingestellt werden, damit LocoTurn auf andere DCC-Adressbereiche reagiert. Der Offset wird auf die jeweiligen DCC-Adressen addiert. Bei manchen Zentralen muss z.B. der Roco-Offset von "-4" hier eingestellt werden, damit LocoTurn die Adressen richtig interpretiert
- Ebenso ist die letzte verwendete DCC Adresse anzupassen, wenn man nicht alle vorgesehenen DCC-Adressen benötigt: #define LAST_USED_DCC_ADDR.
- Liste der DCC-Adressen für die Ports anpassen: #define DCC_PORT_ADDR_LIST. Die Liste muss PORT_CNT-Einträge enthalten. Wenn der Märklin-Modus (DCC_MAERKLIN_7687_COMPATIBLE) verwendet wird, reichen PORT_CNT / 2 Einträge.
- Das #define ADVANCED_SIGNAL_CONTROL = 1 steuert die Signale in Abhängigkeit, ob die DS bei Stillstand mit der jeweiligen Seite an einem aktiven/gültigen oder blinden Port steht. Ob es sich um einen aktiven oder inaktiven/blinden Port handelt, legt das #define PORT_TYPE fest.
- Zur Kalibrierung kann man den DEBUG-Mode einschalten, damit im seriellen Monitor die Werte ausgelesen werden können: #define ENABLE_DPRINTF 1. Da das Serial.print viel Speicher benötigt, muss man ggf. vorübergehend DCC abschalten (#define USE_DCC 0).
- Bei Bedarf Einstellungen der Ausrichtung bzw. Drehrichtung von Drehscheibe, Potentiometer, Dreh/Drückknopf und Display vornehmen.
- Ggf. Änderungen an den Einstellungen der verschiedenen Drehgeschwindigkeiten vornehmen.

Die Abbildung zeigt eine Drehscheibe mit 48 möglichen Ports, davon werden 24 genutzt:

- Die orange-farbigen Ports können später von LocoTurn angefahren werden, sie werden mit der entsprechenden Nummer von 1 - 24 fortlaufend durchnummeriert und im OLED angezeigt.
- Die grün-markierten Ports müssen in das #define **PARTLY_USED_PORTS_TAB** eingetragen werden und zwar genau mit den Nummern aus der Abbildung. Daraus berechnet LocoTurn dann die korrekte Lage des Ports.
- Die hellgrauen Ports sind inaktiv und werden später bei Bewegungen übersprungen und können nicht angefahren werden.



Die kleinste sinnvolle Anzahl Ports ist 2: Ein Gleisabgang plus der dazugehörige gegenüberliegende U-





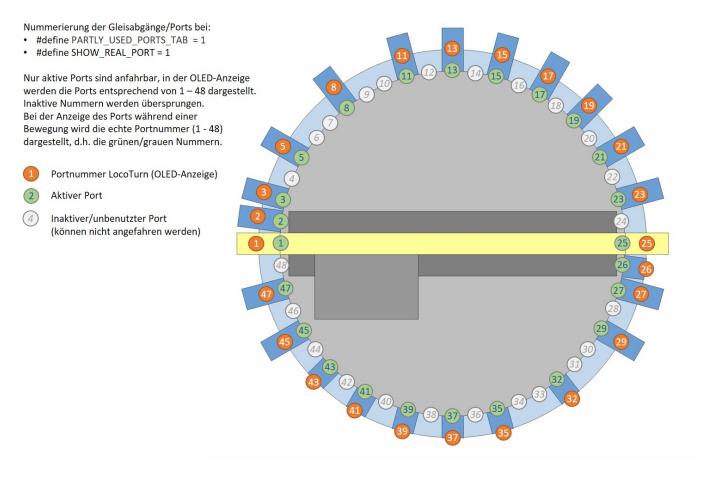
Nummerierung der Ports/Gleisabgänge

In der Software-Version V1.0 hängt die Nummerierung der Ports im wesentlichen vom #define **PARTLY_USED_PORTS** ab. Es werden entweder alle Drehscheiben-Raster mit einer Nummer versehen und sind anfahrbar oder nur aktive Ports erhalten eine Nummer und nur diese aktiven Ports sind anfahrbar.

In der Version V1.1 gibt es darüber hinaus folgende Möglichkeiten, die über das neue #define **SHOW_REAL_PORT** eingestellt werden:

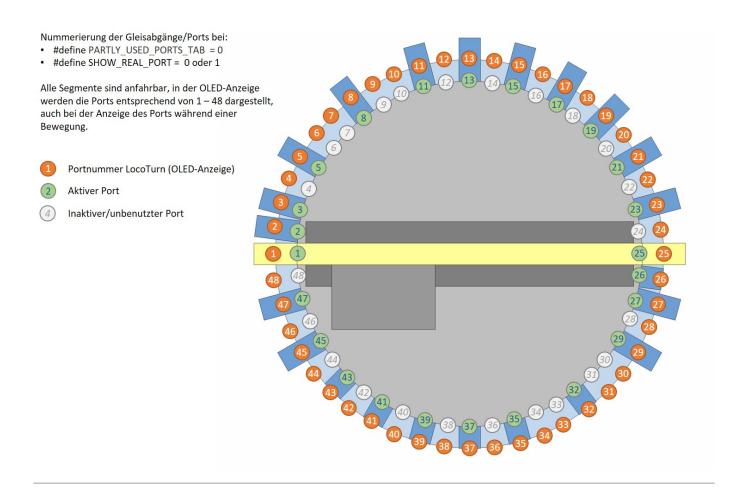
1. **Nur aktive Ports werden in die Nummerierung einbezogen**. Die Nummern in unserer Beispieldrehscheibe laufen von 1 - 24, nicht von 1 - 48. Das entspricht der Version 1.0 oben.

2. **Echte Drehscheiben nummerieren häufig alle Ports durch**, auch inaktive Ports, an denen keine Gleise angeschlossen ist, erhalten eine Nummer. Die Drehscheibe hat demzufolge dann 1 - 48 Ports, von denen nur bestimmte Ports genutzt werden.



3. Wenn alle potentiellen Drehscheibenabgänge angefahren werden können, sieht die

Nummerierung folgendermaßen aus:



Beispielkonfiguration

Ab der Version V1.1 sind einige #defines dazugekommen oder wurden umbenannt. Die folgende Turntable config.h ist nur mit der entsprechenden ino-Version (1.1 oder höher) lauffähig.

Am besten man orientiert sich an den Einstellungen unten und passt alles an die eigene Drehscheibe an. Zu den meisten Parametern gibt es eine Erläuterung, ergänzt durch die jeweilige Kapitel im Wiki:

```
default-Wert aus dem Hauptsketch verwendet!
   Hier stehen nur die wichtigsten Parameter, die am häufigsten
benutzerindividuell eingestellt werden müssen.
   Intention: Bei eine neuen Version des Hauptprogrammes bleiben die
Einstellungen aus diesem Reiter erhalten!
*/
_______
// Parameterset 1: Wantai-Stepper oder StepperOnline-Stepper ohne Getriebe,
400 Steps pro Umdrehung, 6400 Micro-Steps
// 24 Gleisabgänge (= Ports) domapi-Anlage
______
// Zentrale #defines zum Ein-/Ausschalten von Funktionen, um evtl. Speicher
für Tests zu sparen!
//-----
#define USE DCC
                                                 // 1 = use DCC
interface
                0 = disable all DCC routines
#define USE SOUND
                                                 // 1 = Sound
features enabled 0 = Sound features disabled
                                       0
#define USE_SERIAL_INPUT
                                                 // 1 = Serial
monitor input activated 0 = no processing of serial commands --> sollte
in finaler Version ausgeschaltet sein, da man auf der Anlage später nicht
über den PC steuert
#define USE VERBOSE
                                                 // 1 = ausführliches
                                       0
           0 = gekürztes Menü / nur wichtigste Befehle
Menü
#define USE OLED
                                       1
                                                 // 1 = use an OLED
Display (could be disabled ("0") to save memory for tests ~6200 Bytes FLASH
!!!). The outputs are sent to RS232 (serial monitor), then.
#define USE BUTTONS
                                       1
                                                    // Nutzung der 4
Taster auf der roten Panelplatine über analogen Eingang AO; es können auch
zwei Taster gleichzeitig gedrückt sein
#define SHOW STATUS
                                       1
                                                 // 1 = Status-Screen
im Menü aufrufbar, zeigt einige aktuelle Infos (Position, Ports,
Steps/Umdrehung usw.) 0 = nicht aufrufbar, spart Speicher
```

```
/* 0 = Drehen des
#define USE POTI
Poti löst keine Drehscheibenbewegung aus 1 = Poti für manuelle
Bewegungen wird benutzt
                                                    Achtung: Falls
kein Poti angeschlossen wird, ist dieser Wert auf 0 zu setzen. Andernfalls
schleicht die Drehscheibe u.U. vor sich hin.*/
//-----
-----
// *** Debugging***
//-----
#define ENABLE DPRINTF
                                                 // Debug Ausgaben
                                       0
ein, zu Beginn evtl. auf 1 lassen, damit man sich die gespeicherten
Positionswerte im seriellen Monitor ansehen kann
// Typische Konfigurationen:
// ===========
// Normalbetrieb auf der Anlage
//#define USE DCC
                                                 // 1 = use DCC
                0 = disable all DCC routines
interface
//#define USE SOUND
                                                 // 1 = Sound
                0 = Sound features disabled
features enabled
//#define USE SERIAL INPUT
                                                 // 1 = Serial
monitor input activated 0 = no processing of serial commands --> sollte
in finaler Version ausgeschaltet sein, da man auf der Anlage später nicht
über den PC steuert
//#define USE VERBOSE
                                       0
                                                 // 1 = ausführliches
           0 = gekürztes Menü / nur wichtigste Befehle
//define SHOW STATUS
//#define ENABLE DPRINTF
                                                 // Debug Ausgaben
aus
// Test der Steuerung und Einstellung der DS
//#define USE DCC
                                                 // 1 = use DCC
                0 = disable all DCC routines
interface
//#define USE SOUND
                                                 // 1 = Sound
features enabled
                0 = Sound features disabled
//#define USE SERIAL INPUT
                                                 // 1 = Serial
monitor input activated 0 = no processing of serial commands --> sollte
in finaler Version ausgeschaltet sein, da man auf der Anlage später nicht
über den PC steuert
//#define USE VERBOSE
                                                 // 1 = ausführliches
                                       1
           0 = gekürztes Menü / nur wichtigste Befehle
//#define SHOW_STATUS
```

<pre>//#define ENABLE_DPRINTF ein</pre>	1	// Debug Au	sgaben
//			
<pre>// Steuerung Beleuchtungseffekte //</pre>			
#define LEDS_ON_BOARD Beleuchtungssteuerung über die LEDs auf der #define WS281X_BOARD eine kleine Bühnenplatine #define MOBALEDLIB die MLL	1 Platine (ohn 2	e WS28xx, o	// hne MLL) // über
	3		// über
<pre>#define LIGHT_CONTROL MOBALEDLIB //WS281X_BOARD //LEDS_ON_BOARD</pre>	WS281X_BOARD		//
<pre>#define HOUSE_LIGHT_MODUS für direktes WS281x-Board: 0 = einfaches Ei Neonflicker</pre>	2 n/Aus 1 = Fa	de in/out	// nur 2 =
//			
// *** Turntable *** //			
<pre>/* Folgende Einstellungen sind für die Por wichtigsten:</pre>	J		
- Wenn man PARTLY_USED_PORTS auf 1 setz PARTLY_USED_PORTS_REFERENCE ausgewählt und sortiert in die Liste PARTLY_USED_PORTS_TAB und zwar die jeweiligen Nummern der D - Für die Signalsteuerung ist pro genut	- die Portnumme eingetragen rehscheiben-A	ern aufsteig werden, ubgänge	end
PORT_TYPE einzutragen. - Weiterhin muss die Polarisierung unter als von bis Wert POLARISATION_CHANGE_PORT_STPOLARISATION_RELAIS_LIST mit der gewünschter - Die genutzten DCC-Adresse müssen defi	n eingestellt TART bzwEN n Polarisieru	werden. En ID oder als Ing.	tweder
<pre>- Weitere Infos: https://wiki.mobaledlib.de/anleitungen/baua urn_parameter */</pre>	nleitungen/lo	coturn_v10/	150_locot

```
#define PORT CNT
                                            48
Gesamtanzahl Ports/Gleisabgänge, hier im Beispiel eine symmetrische
Fleischmann-Drehscheibe mit 48 Ports im 7,5° Raster
                                                                     // 0 =
#define PARTLY USED PORTS
alle Ports (hier 48) können angefahren werden, 1 = nur gültige Ports (aktive
und passive) können angefahren werden (hier die 24 in der Liste
PARTLY_USED_PORTS_TAB)
#if PARTLY USED PORTS
                                                                     /* If
not all ports of the turntable should be selectable the following table
could define the used ports. The used ports are defined as a reference to a
fully defined table*/
#define PARTLY USED PORTS REFERENCE
                                            48
                                                                     // which
has PARTLY_USED_PORTS_REFERENCE ports (Gesamtanzahl Ports der FLM
Drehscheibe = 48 entspricht 7,5° Raster)
#undef PORT CNT
                                                                     // The
number of real used ports is given in PORT CNT.
#define PORT CNT
                                            24
                                                                     // wir
nutzen nur 24 von 48 Gleisabgängen/Ports
// die folgende Tabelle enthält alle genutzten Ports (aktive und passive),
die man später anfahren kann.
// nicht enthaltene Portnummern (= inaktive Ports) können nicht angefahren
werden und werden bei Drehungen einfach überprungen!
// die Nummern sind i.d.R. nicht fortlaufendend, sondern geben die relative
Position der Ports zur Gesamtanzahl Ports an.
// hier ist alles auf 48 Ports bezogen, davon werden 24 genutzt
#define PARTLY USED PORTS TAB
                                        { 1, 2, 3, 5, 8, 11, 13, 15, 17, 19,
21, 23, 25, 26, 27, 29, 32, 35, 37, 39, 41, 43, 45, 47 }
                                                                // The table
must have PORT CNT entries which are sorted in ascending order
#define SHOW REAL PORT
                                            0
                                                                     // 0 =
im OLED wird bei PARTLY_USED_PORTS = 1 die fortlaufende Portnummer der
genutzten Ports angezeigt 1 = Anzeige des echten Ports (bei FLM-DS 1 -
48), Wert aus dem obigen Array
#endif
#define CIRCUMFERENCE
                                            980.15
                                                                     // 312
mm * Pi = circumference of the turntable [mm]; hier eine FLM-Drehscheibe mit
312 mm Bühnen-Länge
#define OFFSET_PORT_POSITION
                                                                    // Um die
Feinjustierung zu vereinfachen, wird bei Ports mit Defaultwerten die
Position um diese Anzahl Steps entfernt angefahren (CW negativ, CCW
positiv); nur bei unsymmetriwchen DS nutzen!
#define OFFSET PORT POSITION U TURN
                                                                    //
parameter used for U-turns
```

```
/*
   Tipp:
   Man kann das Raster, in dem die Ports initial angelegt werden auch
kleiner machen, z.B. 0,1 Grad genau. Hierzu muss man folgendes eintragen:
   #define PORT CNT
                                            48
Wert hier ist eigentlich egal, er wird weiter unten "überschrieben"
   #define PARTLY USED PORTS
                                            1
                                                                    // 1 =
only valid ports can be used
   #if PARTLY USED PORTS
    #define PARTLY_USED_PORTS_REFERENCE
                                            3600
                                                                    //
Einteilung der DS in 3600 Ports
    #undef PORT CNT
    #define PORT CNT
    #define PARTLY USED PORTS TAB
                                            {300, 1350, 2100, 3150} // davon
nutzen wir nur 4 Ports an diesen Positionen
   #endif
   Im Prinzip teilt das den Vollkreis in 3600 Ports/Gleisabgänge, von denen
aber nur 4 genutzt werden, die quasi an beliebiger Stelle liegen können. Das
Beispiel oben hat 4 anfahrbare Ports bei:
   Grad
                           Uhrzeit
    30°
                              1
   135°
                           "halb" 5
   210°
   315°
                           "halb" 11
*/
// *** Signalsteuerung ***
#define ADVANCED SIGNAL CONTROL
                                            1
                                                                   // 1 =
Steuerung der Signale in Abhängigkeit, ob die DS-Seite bei Stillstand an
einem aktiven/gültigen oder passiven/blinden Port steht. 0 = Einfache
Signalsteuerung.
// Option 1: wenn PARTLY USE PORTS = 0
// Wenn man alle Ports anfahren können soll, hie z.B. 48, dann brauchen wir
eine Liste mit 48 Nullen und Einsen
// Gleisabgang
                                            1 2 3 4 5 6 7 8 9 10 11
12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36
37 38 39 40 41 42 43 44 45 46 47 48
```

```
// #define PORT TYPE
                                           1, 1, 1, 0, 0, 1, 0, 1, 0,
1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0 // Port-Type = 1 if there is an
active / valid railway connected, 0 = passive/blind port (no railway
connected). The table must have PORT CNT entries. Required for advanced
signal control (signal = red if turntable is no at a valid port)
// Option 2: wenn PARTLY USE PORTS = 1
// als Beispiel hier: bei 24 Ports sind einige aktiv, einige sind blind! Wir
brauchen 24 Nullen und Einsen
// Gleisabgang
                                           1, 2, 3, 5, 8, 11, 13, 15, 17,
19, 21, 23, 25, 26, 27, 29, 32, 35, 37, 39, 41, 43, 45, 47
#define PORT TYPE
                                           1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0 // Port-Type = 1
if there is an active / valid railway connected, 0 = blind port (no railway
connected). The table must have PORT_CNT entries. Required for advanced
signal control (signal = red if turntable is no at a valid port)
// *** Polarization relays ***
#define POLARISATION RELAIS PIN
                                           Α1
Polarisations-Relais for dual rail system (Set to -1 if not used)
#define POLARISATION RELAIS INVERS
                                                                    // 0:
                                          0
Pin is set to high, if POLARISATION_RELAIS_LIST[Port] = 1 1: Pin is
set to low. if 0
#define POLARISATION CHANGE PORT START
                                           11
#define POLARISATION CHANGE PORT END
                                           22
/* Teilt die Drehscheibe für die Polarisierung quasi in 2 Hälften.
   Ab dem hier definierten START-Port (externe Nummerierung) wird das
Polarisationsrelais bis zum i.d.R. gegenüberliegenden END-Port
eingeschaltet.
   Die hier eingegebenen Zahlen müssen zwischen 1 und PORT_CNT liegen!
START muss < END sein!
   Die eine Hälfte der Scheibe ist Polung 1, die andere Polung 2.
   Spart einige Bytes ;-)
    -1 = Polarisation gemäß der unten stehenden Liste
*/
//#define POLARISATION RELAIS LIST
                                             0, 1, 0, 1 , 0, 1, 0, 1 ...
// Pro genutztem DS-Port hier ein Eintrag in der Liste mit 0 oder 1
```

```
// *** Direction and control settings ***
//-----
                                                                 /* Set
#define ROTATIONSWITCH DIRECTION
from 1 to -1 to change the direction of the rotation switch
                                                                     -1 =
Drehen nach rechts höhere Port-Nr.; links = niedrigere Port-Nr.*/
#define USE ROTARY ENCODER MOVEMENT
                                                                  // 0 =
Drehen des Encoders löst keine Drehscheibenbewegung aus, Menüs fkt. aber
1 = Encoder steuert die DS und Menü fkt.
#define SPEED POTI DIRECTION
                                                                  // Set
to -1 to change the direction of the speed poti; bei domapi-Platine "-1"
#define ROTATIONSWITCH MENU DIR
from 1 to -1 to change the direction of the rotation switch in the menu:
Rechts-Drehung = Zeile nach unten im Menü
#define TURNTABLE DIRECTION
                                                                  // Set 1
to -1 to change the rotation / port number direction, abhängig von der
Polung der Motorspulen --> einfach ausprobieren!
// Wenn fahrtdauerabhängige Sounds verwendet werden, dann muss ENCODER LOGIC
= 1 sein, sonst passen die Sounds nicht zur Bewegung ;-)
#define ENCODER LOGIC
                                                                     1 =
Drehencoder drehen während Stillstand ermöglicht Portauswahl
- Bewegung wird erst gestartet, wenn Encoder-Taste gedrückt wird
- während der Bewegung kann die anzufahrende Portnummer nicht mehr geändert
werden
- Taste während Bewegung ruft das Menü auf und stoppt die Bewegung
Encoder startet Bewegung, während Bewegung kann weiter gedreht werden*/
#define DIRECTION CHANGE POSSIBLE
                                          0
                                                                  // 1 =
Richtungsänderung erlaubt, wenn anderer Befehl bei bereits gestarteter
Bewegung kommt 0 = keine Richtungsänderung, eine einmal eingeschlagene
Richtung wird beibehalten
#define MOVE STATUS INVERS
                                          1
                                                                  /*
Invert the level of the S88 MOVING PIN:
                                                                     - 1
and turntable moving = Optokoppler sperrt.
                                                                     - 1 +
Stillstand: OK steuert durch und erzeugt eine Rückmeldung.
                                                                     Viele
Steuerungssoftwarepakete erwarten ein Signal, wenn die DS steht (d.h.
```

```
angekommen ist)*/
          -----
// *** Schrittmotor ***
//-----
#define FIXED STEPS PER ROUND
manuelle Vorgabe der Anzahl Steps pro Umdrehung; "0" = automatische
Ermittlung
#define FIXED STEPS HAS CONTACT
                                         0
                                                               //
manuelle Vorgabe des Getriebespiels; "-1": dann wird das Spiel automatisch
ermittelt, Werte >= 0 --> manuelle Vorgabe
#define NOT ENABLE PIN
                                         6
                                                                /* Set
to -1 if the stepper driver has an automatic power mode like the TMC2100
                                                                   The
pin of the module must be left open (std 6)
                                                                   Use
Pin 6 for TMC2208 */
#define ENABLE ALWAYS ON
Vermeidet bei meinem aktuellen Steppermotor ohne Getriebe das Ruckeln in der
Start-/Endposition
Nachteil: Motor wird dauerhaft angesteuert und wird warm!
                                                                   Set
to 1 for a powerful stepper without gearbox because the magnetic field will
change the position when powered of
                                                                   If a
stepper driver like the TMC2100 is used it's better to disable the
NOT ENABLE PIN (-1) to activate the automatic power saving mode instead.
                                                                   Bei
Einsatz TMC2208 sollte man den Wert 1 verwenden! */
#define STEPPER RAMP LENGTH
                                         180
                                                                /* Steps
to speed up the stepper to prevent loosing steps
                                                                   Set
to 50 if 1/16 steps are used (MS1 - MS3 connected do +5V)*/
#define MOVE SPEED1
                                         600
                                                                //
Default speed and activated when DCC SET SPEED1 ADDR is received
#define TURNBACK SPEED
                                         600
                                                                // Speed
used for TurnBackAndSetZero
#define USE TURNBACK
                                                                // 1 =
Turn back & set 0-point will be executed if 0-point is detected in negative
rotation direction 0 = not used
```

#define MOVE_SPEED2	400	// Speed		
activated when DCC_SET_SPEED2_ADDR is #define MOVE SPEED3	received 200	// Speed		
activated when DCC_SET_SPEED3_ADDR is		// Speed		
#define MOVE_SPEED4	2600	// Speed		
activated when DCC_SET_SPEED4_ADDR is	received			
#define CLEARANCE_TEST_SPEED	1200	//		
Speed used in the clearance test				
#define CALIBRATE_SPEED_FAST	1200	//		
Speed used for the zero point and tota #define CALIBRATE SPEED SLOW	200	ection // Slow		
speed used for the advanced zero point		,, 5.0		
#define ZERO_DETECTION_OFFSET	60	// Used		
for moving a little bit away from Hall	Sensor			
#define MIN STEPS HAS CONTACT	0	/*		
Typical value = 30.		·		
move fact to the centact point in the	"Doti" mode if Ctons	Don't		
move fast to the contact point in the "Poti" mode if Steps_Has_Contact is				
below this value. Set to 0 to disable the check (Always move fast to the				
contact point)				
Problem: The Steps_Has_Contact detecti	on may be not accurate	e because of		
sensor errors		0 =		
kein Anfahrruckler		TODO		
TODO: evtl. muss die Routine, die das Getriebespiel "überbrückt" direkt in der				
Loop aufgerufen werden, wenn die eigentlich Bewegung gestartet wird*/				
	_			
#define ALWAYS_CALIBRATE_AT_START	1	// Set		
to 1 to calibrate the 0-position (Hallsensor) every time when the program is started. Helps to compensate little jumps of the turntable when turned on.				
	•			
#define ADVANCED_REFERENCE_DETECTION	1	// "1"		
fährt den Hallsensor beim 0-Punkt Kalibieren schnell an, dreht ein Stückchen zurück und fährt ihn nochmals langsam an und speichert die Referenzposition				
zaraek ana ram a zim neemiate tangsam	an and sporence are .	10.0.0.0.12001		
//				
//				
// *** OLED ***				
//				
#define OLED TYP	13	//		
Tested with the following displays 87,				
#define USE_u8x8	1	// 1 bei		
96 und 13				

```
#define SHOW POSITION MAINSCREEN
                                                       //
Positionsanzeige auf dem Hauptscreen
//-----
// ** Flashing & light setup ***
//-----
#define MOVING FLASH INVERS
                                   0
                                                       //
Normal: 0 = LED connected to GND
#define MOVING FLASH MODE
                                   2
                                                       // 1 =
Blink, 2 double flash
#define HOUSE BRIGHTNESS
                                   255
Brightness of house LED, valid values 0 - 255, 0 = off, 255 = very bright,
saves separate resistor ;-)
______
// *** sketch internal parameters ***
//-----
#define BITSCHIEBER
                                   1
                                                        // 1 =
direkte Bitmanipulation 0 = digitalRead/Write, pinMode, FastPin ToDo:
Prüfen, ob irgendwelche Libs noch digitlaRead etc. nutzen
#define SPEED POTI MID RANGE
                                                        //
Range (+/-) of the speed poti which is considered as 0
#define SPEED POTI CENTER
                                                        //
Center position of the speed poti (Normally 512)
#define ANALOG SPEED DIVISOR
                                   40
Divisor used to calculate the analog speed with the poti (std 8, 50)
                                                        // 1 =>
#define MIN ANALOG SPEED
                                   40
10 sec reaction time ;-( Reaction time = 10 sec / MIN_ANALOG_SPEED
#define ALWAYS_CHECK_STEPS_ONE_TURN
                                                        //
Always check the steps for one turn at power on
#define ASK TO UPDATE ALL
                                                        // = 1:
Ask to update all ports when Port 1 is redefined > 1: Ask to update all
ports for every port
#define ALWAYS SET ZERO IN POS DIR
                                                        // 1 =
                                   1
Check and adjust the O-position always, if the hall sensor has been detected
0 = only if 1 complete turn was moved
```

```
//-----
// *** Sound Setup ***
//-----
#define DFPLAYER
                                                                 // mit
                                          1
DS-Kartenslot
#define J06500
                                          2
                                                                 // 2MB
Speicher onboard
#define USE SOUNDMODULE
                                          DFPLAYER //J06500
                                                                 //
Auswahl des verwendeten Soundmoduls
#define USE SOUND SERIAL
                                         -1
                                                                 //
SMART_JQ6500\_SERIAL_PIN --[1K] -- TX; -1 = hardware serial is used (TX-Pin)
for JQ6500 interface; DFPlayer uses softwareserialTX on Pin D12
#define SOUNDMODULE VOLUME
                                         18
                                                                 //
Range: 0..30 (-1 = Don't change the volume)
#define DELAY TURN START SOUND
                                         7700
Delay before start moving if sound is played, hängt ab vom verwendeten
soundfile; bei mir machen 7,7s Sinn!
                                                                    Bei
Verwendung eines DFPlayers ca. 300 ms dazu addieren, da dieses Soundmodul
verzögert startet über softwareserial*/
#define SOUND1 FILENR
                                                                 //
sound-file number of JQ6500 / DFPLayer for turntable start and running
(Hupe, Anfahren und Drehen)
#define SOUND2 FILENR
                                                                 //
sound for turntable stop (aktuell nur die Hupe)
#define SOUND CONTINUOUS MOVE
                                                                 // 0 =
kein Abspielen von sounds bei DS-Bewegungen mit dem Poti 1 = Sound
wird automatisch abgespielt
#define FAHRTDAUER MESSEN
                                          0
#define ADVANCED SOUND
                                          1
es werden zwei Sounds abgespielt, einer beim Starten der Bewegung (gesampelt
aus Hupen, Anfahren und Fahren), einer beim Stoppen (Hupen); Bremssound ist
nicht verfügbar
die Fahrtdauer wird berechnet und ein entsprechender langer Sound gestartet,
hierfür brauchen wir mehrere Files auf dem Soundmodul.
Jedes file entält Hupen, Anfahren, Fahren, Bremsen bis zum Stopp und
abschließendes Hupen. Der variable Anteil ist das Fahren, das muss solange
hintereinander kopiert werden, dass es für die entsprechende Fahrtstrecke
reicht
                                                                    Ein
separater Sound beim Stoppen (Hupen) wird nur bei manuellem Stopp
abgespielt, beim auto. Fahren ist der sound bereits im Fahrgeräusch
```

```
enthalten
ENCODER LOGIC dann auf 1 setzen (siehe oben)*/
#define DCC SOUNDFILE 1
                                          1
                                                                //
File-Nr. auf dem Soundmodul, das abgespielt wird mit dem entsprechenden DCC-
Befehl
#define DCC SOUNDFILE 2
                                          2
J6500; Dateien stehen im Rootverzeichnis, Reihenfolge geht nach Reihenfolge
des Kopierens auf das Modul
#define DCC SOUNDFILE 3
                                                                 //
DFPlayer: Dateien müssen im \mp3-Folder stehen, Nomenklatur 0001_beliebiger
Text (4-stellige Nummer, führende Nullen + sprechender Text)
#define DCC SOUNDFILE 4
                                          28
#define DCC_SOUNDFILE_5
                                          29
#define DCC SOUNDFILE 6
                                          30
// *** Pins ***
//-----
// n/a --> alle aus dem Hauptsketch verwenden!
// *** DCC ***
//-----
#define DCC OFFSET
                                                    // -4 = Roco-Offset
für Roco-Zentrale, 0 = DCC addresses start at #211, using this parameter
you can use other DCC ranges in case of using several turntables with one
central station or you rewuire other address ranges
#define DCC WAITING TIME
                                         350
                                                    // waiting time [ms]
for further DCC-tickets before carrying out the DCC command
#define DCC MAERKLIN 7687 COMPATIBLE
                                         1
                                                    // 1 = DCC-Befehle
nur für den 1. Halbkreis der DS verwenden. In Verbindung mit der
Drehrichtung kann trotzdem jeder Port so angefahren werden, dass das DS-Haus
richtig steht, notwendig bei Verwendung einer PC-Steuerung!
                                                    // 0 = wir berauchen
für alle Ports DCC-Adressen um sie anfahren zu können ==> wird normalerweise
bei Anlagen ohne PC-Steuerung verwendet!
#if DCC_MAERKLIN_7687_COMPATIBLE == 1
   // sollte man nur verwenden, wenn man mit einer PC-Steuerung arbeitet
```

```
#define LAST USED DCC ADDR
                                            DCC CHKADDR((234 + DCC OFFSET),
      // die "234" und das "GRN" sind an die tatsächlich verwendete letzte
Adresse unten anzupassen!
#else
    // wenn man ohne PC-Steuerung arbeitet
    #define LAST USED DCC ADDR
                                            DCC CHKADDR((229 + PORT CNT / 2
- 1 + DCC OFFSET), GRN) // wird automatisch berechnet
#endif
// Momentan 24 (von 48) genutzte Ports = { 1, 2, 3, 5, 8, 11, 13, 15, 17,
19, 21, 23, 25, 26, 27, 29, 32, 35, 37, 39, 41, 43, 45, 47 }
#if DCC MAERKLIN 7687 COMPATIBLE == 1
// sollte man nur verwenden, wenn man mit einer PC-Steuerung arbeitet
// Es werden nur für die Ports im ersten Halbkreis DCC-Adressen benötigt
// Bei 24 Ports braucht man hier also 12 DCC-Adressen (= 12 Zeilen)
// Die erste Zahl in der Klammer ist immer fortlaufend und startet bei 1.
Sie hat nichts mit der Portnummer oben zu tun (die kann nicht-fortlaufend
sein).
// Pro DCC-Adresse gibt es 2 Befehle bzw. Richtungen: RED und GRN
                                        DCC PORT ADDR(1, (229 + DCC OFFSET),
#define DCC PORT ADDR LIST
RED), \
    DCC PORT ADDR(2, (229 + DCC OFFSET), GRN), \
    DCC PORT ADDR(3, (230 + DCC OFFSET), RED), \
    DCC PORT ADDR(4, (230 + DCC OFFSET), GRN), \
    DCC_PORT_ADDR(5, (231 + DCC_OFFSET), RED), \
    DCC PORT ADDR(6, (231 + DCC OFFSET), GRN), \
    DCC_PORT_ADDR(7, (232 + DCC_OFFSET), RED), \
    DCC PORT ADDR(8, (232 + DCC OFFSET), GRN), \
    DCC PORT ADDR(9, (233 + DCC OFFSET), RED), \
    DCC_PORT_ADDR(10, (233 + DCC_OFFSET), GRN), \
    DCC PORT ADDR(11, (234 + DCC OFFSET), RED), \
    DCC_PORT_ADDR(12, (234 + DCC OFFSET), GRN)
    // die vorhergehende Zeile beschreibt die letzte Adresse und muss oben
ins #define LAST USED DCC ADDR eingetragen werden
#else
// wenn man ohne PC-Steuerung arbeitet
// DCC-Adressen für den kompletten Vollkreis notwendig!
// Bei 24 Ports braucht man hier auch 24 DCC-Adressen (= 24 Zeilen)
#define DCC PORT ADDR LIST
                                        DCC PORT ADDR(1, (229 + DCC OFFSET),
RED), \
    DCC PORT ADDR(2, (229 + DCC OFFSET), GRN), \
    DCC PORT ADDR(3, (230 + DCC OFFSET), RED), \
    DCC_PORT_ADDR(4, (230 + DCC_OFFSET), GRN), \
    DCC PORT ADDR(5, (231 + DCC OFFSET), RED), \
    DCC PORT ADDR(6, (231 + DCC OFFSET), GRN), \
    DCC_PORT_ADDR(7, (232 + DCC_OFFSET), RED), \
    DCC PORT ADDR(8, (232 + DCC OFFSET), GRN), \
```

```
DCC PORT ADDR(9, (233 + DCC OFFSET), RED), \
   DCC PORT ADDR(10, (233 + DCC OFFSET), GRN), \
   DCC PORT ADDR(11, (234 + DCC OFFSET), RED), \
   DCC_PORT_ADDR(12, (234 + DCC_OFFSET), GRN), \
   DCC PORT ADDR(13, (235 + DCC OFFSET), RED), \
   DCC_PORT_ADDR(14, (235 + DCC_OFFSET), GRN), \
   DCC PORT ADDR(15, (236 + DCC OFFSET), RED),
   DCC PORT ADDR(16, (236 + DCC OFFSET), GRN), \
   DCC PORT ADDR(17, (237 + DCC OFFSET), RED), \
   DCC_PORT_ADDR(18, (237 + DCC_OFFSET), GRN),
   DCC PORT ADDR(19, (238 + DCC OFFSET), RED), \
   DCC PORT ADDR(20, (238 + DCC OFFSET), GRN),
   DCC PORT ADDR(21, (239 + DCC OFFSET), RED), \
   DCC PORT ADDR(22, (239 + DCC OFFSET), GRN), \
   DCC PORT ADDR(23, (240 + DCC OFFSET), RED), \
   DCC_PORT_ADDR(24, (240 + DCC_OFFSET), GRN)
// Bei mehr als 41 Einträgen/Zeilen in dem obigen #define spinnt der
Compiler!
// Workaround: dann müssen die zusätzlichen Zeilen in das #define ... LIST 2
aufgenommen werden ;-)
   #define DCC PORT ADDR LIST 2
                                         DCC PORT ADDR(27, (242 +
DCC OFFSET), RED), \
   DCC_PORT_ADDR(28, (242 + DCC_OFFSET), GRN), \
   DCC PORT ADDR(29, (243 + DCC OFFSET), RED), \
   DCC PORT ADDR(30, (243 + DCC OFFSET), GRN),
   DCC PORT ADDR(31, (244 + DCC OFFSET), RED), \
   DCC PORT ADDR(32, (244 + DCC OFFSET), GRN), \
   DCC PORT ADDR(33, (245 + DCC OFFSET), RED),
   DCC PORT ADDR(34, (245 + DCC OFFSET), GRN), \
   DCC PORT ADDR(35, (246 + DCC OFFSET), RED),
   DCC PORT ADDR(36, (246 + DCC OFFSET), GRN), \
   DCC_PORT_ADDR(37, (247 + DCC_OFFSET), RED), \
   DCC PORT ADDR(38, (247 + DCC OFFSET), GRN), \
   DCC PORT ADDR(39, (248 + DCC OFFSET), RED), \
   DCC PORT ADDR(40, (248 + DCC_OFFSET), GRN),
   DCC PORT ADDR(41, (249 + DCC OFFSET), RED), \
   DCC_PORT_ADDR(42, (249 + DCC_OFFSET), GRN), \
   DCC_PORT_ADDR(43, (250 + DCC_OFFSET), RED),
   DCC PORT ADDR(44, (250 + DCC OFFSET), GRN), \
   DCC PORT ADDR(45, (251 + DCC OFFSET), RED), \
   DCC PORT ADDR(46, (251 + DCC OFFSET), GRN), \
   DCC_PORT_ADDR(47, (252 + DCC_OFFSET), RED),
   DCC PORT ADDR(48, (252 + DCC OFFSET), GRN)
*/
#endif
// ggf. hier weitere Zeilen ergänzen, wenn zusätzliche DCC-Befehle für
```

```
weitere Ports benötigt werden
// die letzte DCC-Adresse muss sich oben aus der #define-Formel bei
LAST USED DCC ADR ergeben !!!
//-----
// *** Tastenauswertung ***
// Das #define definiert die 4 Schwellwerte für die 4 Taster auf der
Panelplatine hier in der Config-Datei
// Bei Änderungen in der ino-Datei bleiben die individuellen Werte unten
erhalten!
#define BUTTON THRESHOLDS 970, 327, 696, 872
#define HOME RUN PORT 24
                               // wird bei der Tastersteuerung verwendet,
bei einem entsprechenden Tastendruck fährt die DS zum angegebenen Port
/*
  folgende Aktionen stehen für die Taster zur Verfügung:
   B Toggle House()
                                Hausbeleuchtung ein/aus
   B_Toggle_Sound()
                                Sound ein/aus
   B Toggle Signal House()
                                Signal Hausseite rot/weiß
   B_Toggle_Signal_Opposite()
                                Signal Gegenüber rot/weiß
   B Signal House red()
                                Signal Haussseite rot
   B Signal House white()
                                dito. weiß
   B_Signal_Opp_red()
                                Gegenüber rot
   B Signal Opp white()
                                dito. weiß
   B U Turn CW()
                                180°-Drehung CW
   B U Turn CCW()
                                180°-Drehung CCW
                                Anfahren der Home-Position
   B Home Run()
   B Toggle Flash()
                                Warnleuchte ein/aus
   Play_Sound(n)
                                Sound n auf dem JQ6500 abspielen
   Auto Calibrate(1)
                                führt die Kalibirierung des Nullpunktes
durch
*/
// Hier den Tastern die Aktion zuordnen (kann man auch mehrfach machen, dann
";" dazwischen!):
// #define BUTTON n ohne weitere Zuordnung deaktiviert den Knopf bzw. löst
keine Aktion aus, z.B. "#define BUTTON 1"
// kurzer Tastendruck:
#define BUTTON 1
                   Play Sound(28) // spielt Soundfile 28 ab; falls
Sound generell deaktiviert wurde, muss man die Taste deaktivieren, siehe
oben
```

From:

https://wiki.mobaledlib.de/ - MobaLedLib Wiki

Permanent link:

Last update: 2024/10/03 07:42

